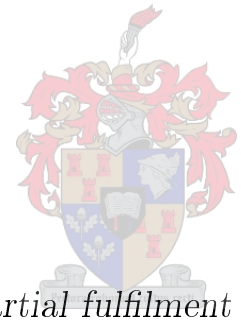
Evaluation of VANET standards and protocols for distributed licence plate detection and reporting

by

Jacobus Christoffel Truter



Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Electronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Mr. A. Barnard

Co-supervisor: Prof. H.A. Engelbrecht

December 2019

BYLAE 1

UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
Jou kennisvennoot • your knowledge partner

Plagiaatverklaring / Plagiarism *Declaration*

1 Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.

2 Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
I agree that plagiarism is a punishable offence because it constitutes theft.

3 Ek verstaan ook dat direkte vertalings plagiaat is.
I also understand that direct translations are plagiarism.

4 Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.

5 Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

Studentenommer / *Student number*	**Handtekening / *Signature***
J.C. Truter **Voorletters en van / *Initials and surname***	26 September 2019 **Datum / *Date***

Abstract

Evaluation of VANET standards and protocols for distributed licence plate detection and reporting

J.C. Truter

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MEng (Electronic)

December 2019

Implementation and evaluation of existing VANET standards and protocols for suitability of distributed licence plate detection and reporting in an urban scenario was the problem this project set out to investigate. Simulation was used to approach the problem since large scale deployment of commercially available VANET hardware was not feasible. A simulation environment providing realistic wireless networking and vehicular traffic patterns with which to evaluate VANET standards and protocols was successfully realised. An open source, Linux-based hardware solution (VANET OBU fully implementing IEEE 802.11p and AODV was created. Core aspects of the network simulation environment was successfully verified and calibrated by means of experimentation with the hardware solution. A distributed licence plate detection application was implemented in the verified and calibrated simulation environment, and analysis of large scale simulations resulted in the conclusion that IEEE 802.11p and AODV is suitable for distributed license plate detection and reporting as per its implementation in this project.

Uittreksel

Evaluasie van bestaande VANET standaarde vir geskiktheid in verspreide nommerplaat herkenning en rapportering

("Evaluation of VANET standards and protocols for distributed licence plate detection and reporting")

J.C. Truter

Departement Elektriese en Elektroniese Ingenieurswese,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.

Tesis: MIng (Elektronies)

Desember 2019

Die hoof probleem wat die projek aangespreek het, is die implementering en evaluasie van bestaande VANET standaarde, om hul geskiktheid te bepaal vir gebruik in verspreide nommerplaat herkenning en rapportering. Simulasie was gebruik om die probleem aan te pak sedert dit nie moontlik was om kommersiële VANET hardeware op groot skaal te ontplooi nie. 'n Simulasie omgewing wat realistiese draadlose netwerk en voertuig-verkeer patrone kan simuleer was suksesvol opgestel en kon gebruik word om die VANET protokolle en standaarde te evalueer. 'n Oop bron Linux-gebasseerde hardeware oplossing (VANET OBU wat IEEE 802.11p en AODV ten volle implementeer, was geskep. Hierdie hardeware was gebruik om die simulasie omgewing suksesvol te verifieer en te kalibreer deur middel van eksperimentasie. Verspreide nommerplaat herkenning was geimplementeer in die geverifeerde simulasie omgewing, en grootskaalse simulasies was uitgevoer. Analise van die resultate het gelei tot die gevolgtrekking dat IEEE 802.11p en AODV geskik is vir verspreide nommerplaat herkenning en rapportering soos in die projek geimplementeer.

Acknowledgements

I would like to express my sincere gratitude to the following people:

Mr. Arno Barnard and Prof Herman Engelbrecht for their continuous guidance, support, and mentorship,

Prof Herman Engelbrecht for financial assistance throughout the duration of my Master's degree,

Pierre Ackermann for the immense amount of motivation, help and time invested in me,

Telectro CC for kindly providing single board computers in the time of need,

My family that were always there during every step of the way, even at times when I was distant,

My other half, my favourite, my perfect puzzle piece, Danielle - for the unending love, caring and support,

And to my God, friend and Saviour without whom this would not be possible.

Dedications

Aan my gesin - Chris, Christel, Estelle en Petrus wat altyd ondersteunend en trots, op als wat ek aangepak en bereik het. En Pierre, wat nooit verder as 'n foon oproep weg was nienie en altyd sy tyd en moeite opgeoffer het.
Aan Danielle, my gunsteling, my lief, my lewensmaat - dankie vir al jou liefde, ondersteuning, tyd en geduld.
Aan my Heer, my God, sonder wie hierdie nie moontlik sou wees nie en my lewe 'n heel ander pad sou volg.

Contents

List of Figures

List of Tables

List of Abbreviations

AC Access Category

ACK Acknowledgement

AES Advanced Encryption Standard

AIFS Arbitration Inter-frame Space

AIFSN Arbitration Inter-frame Space Number

ALPR Automatic License-plate Recognition

AODV Ad-hoc On-Demand Distance Vector

ARIB Association of Radio Industries and Businesses

BE Best Effort

BK Background

BSS Basic Service Set

BTP Basic Transport Protocol

CAM Cooperative Awareness Message

CCH Control Channel

CCM Cypher Block Chaining Message Authentication Code

CW Contention Window

DCC Decentralised Congestion Control

DCF Distributed Coordination Function

DSRC Dedicated Short-Range Communications

EC European Commission

ECDSA Elliptic Curve Digital Signature Algorithm

ECIES Elliptic Curve Integrated Encryption Scheme

ETSI European Telecommunications Standards Institute

EDCA Enhanced Distributed Channel Access

EU European Union

FCC Federal Communications Commission

GPS Global Positioning System

GUI Graphical User Interface

HCCA HCF Controlled Channel Access

HCF Hybrid Coordination Function

ICMP Internet Control Message Protocol

IEEE Institute of Electrical and Electronics Engineers

IP Internet Protocol

ITS Intelligent Transportation Systems

JOSM Java OpenStreetMap Editor

LLC Logical Link Control

LOS Line of Sight

LPDR Licence Plate Delivery Ratio

MANET Mobile Ad-hoc Network

MAC Media Access Control

MIB Management Information Base

MLME MAC Layer Management Entity

MSDU MAC Service Data Unit

MTU Maximum Transmission Unit

NIC Network Interface Card

NLOS Non-Line of Sight

OBU On-board Unit

OCB Outside the Context of a BSS

OFDM Orthogonal Frequency-division Multiplexing

OSI Open Systems Interconnection

OSM OpenStreetMap

PCB Printed Circuit Board

PDR Packet Delivery Ratio

PHY Physical

PCF Point Coordination Function

POI Point of Interest

QoS Quality of Service

RERR Route Error

RM Resource Management

RREP Route Reply

RREQ Route Request

RSU Road Side Unit

RWM Random Waypoint Movement

RX Receive

SCH Service Channel

SIFS Short Inter-frame Space

SNIR Signal-to-Interference-Plus-Noise Ratio

STA Station

TCP Transmission Control Protocol

TX Transmit

TXOP Transmission Opportunity

UDP User Datagram Protocol

USA United States of America

UTC Coordinated Universal Time

US United States

V2I Vehicle-to-Infrastructure

V2V Vehicle-to-Vehicle

VI Video

VO Voice

VANET Vehicular Ad-hoc Network

VANETs Vehicular Ad-hoc Networks

WAVE Wireless Access in Vehicular Environments

WBSS WAVE Basic Service Set

WLAN Wireless Local Area Networks

WSA Wave Service Advertisement

WSM WAVE Short Message

WSMP WAVE Short Message Protocol

Nomenclature

Units of Measurement

Hz	Frequency	[Hertz]
s	Time	[Seconds]
m	Length	[Meters]
b	Basic Unit of Digital Information	[Bit]
B	Unit of Digital Information	[Byte]

Prefixes for units of measure

p	Pico	[10^{-12}]
n	Mega	[10^{-9}]
μ	Micro	[10^{-6}]
m	Mili	[10^{-3}]
c	Centi	[10^{-2}]
k	Kilo	[10^{3}]
M	Mega	[10^{6}]
G	Giga	[10^{9}]

Chapter 1

Introduction

1.1 Motivation and topicality of this work

Crime is a major issue in South Africa, presenting a dire need for preventative measures and aids for criminal investigation. Vehicular crimes such as carjackings and cash-in-transit robberies have seen a significant increase over the past few years, with carjackings increasing by 14.3% in 2016 alone [10]. Having access to up-to-date, or any relevant information during a criminal investigation is essential to identify and track down suspects. Local authorities could greatly benefit by having information such as movement patterns, and sightings of stolen and crime associated vehicles.

Fixed infrastructure licence plate detection systems can be used to obtain such information. However, these systems are usually deployed in locations such as large city centres, major highways and airports resulting in a relatively small area of coverage. Due to the fixed nature of such systems, suspects could potentially avoid these by re-routing.

A mobile, decentralised licence plate detection system is proposed. By installing inconspicuous license plate detection and wireless communication hardware in public transportation, fleet or service vehicles, such a system could be created. En-route, equipped vehicles would identify and log time stamped, license plate and location data. These vehicles will create an ad hoc wireless network amongst one another with which to aggregate all observational data to back-end infrastructure via fixed data collectors placed in specific locations. It would be possible for such a system to operate independently of any pre-existing infrastructure such as mobile networks or existing transportation information networks.

A system like this would have the potential to cover a large area, provide up-to-date location information on vehicles of interest, penetrate locations where placement of fixed cameras are not optimal, be unpredictable to sus-

pects, as well as providing scalability by simply equipping more vehicles with the necessary hardware.

Networking and communication would be critical to the functionality and effectiveness of such a system, thus focus will specifically be placed on the networking aspect during this project. Wireless technologies and standards used in Vehicular Ad-hoc Networks (VANETs) can be used to realise wireless communication between the vehicles. Vehicular Ad-hoc Network (VANET) is a subcategory of Intelligent Transportation Systems (ITS) and is a relatively new field of research, opening up opportunities for new studies and applications such as this envisaged solution. Not much research has been published on non-safety related and standalone application implementation in VANETs, which begs the question if VANET and its related technologies would be suitable to realise an application such as the envisaged solution. ITS has also not seen much implementation in South Africa unlike in European countries, thus providing an opportunity to bring a possible VANET related solution to the table.

1.2 Background

A VANET is a dynamic wireless network consisting of vehicle-mounted wireless devices that performs automatic network configuration and do not rely on pre-existing infrastructure - hence the term 'ad hoc'. Each device can function as a sender, receiver and router, exchanging information with other devices within their reach. Information exchange in a VANET can either be Vehicle-to-Vehicle (V2V) or Vehicle-to-Infrastructure (V2I), with single- or multi-hop data dissemination. VANETs can be seen as a subset of Mobile Ad-hoc Network (MANET), with the main difference being the mobility characteristics of nodes. Unlike MANET nodes (e.g. laptops, cell phones and other smart devices), VANET nodes travel at much higher speeds, have much shorter stationary periods and move along constrained paths like roads and highways.

The main attraction of VANET is its potential use for road safety and traffic management applications. Vehicles will be able to exchange information directly with one another, providing updates to their neighbours such as impact or emergency break notifications, which can aid in management and prevention of potentially harmful situations. Infotainment services, including internet connectivity and media streaming, are also intriguing applications for VANETs.

VANET as a whole is still in its infancy, with Institute of Electrical and Electronics Engineers (IEEE) draft standards for this specific communication environment only being released in 2010. Since this field is relatively new, com-

mercial hardware is still expensive and difficult to obtain. As a result, many researchers are turning to theory and simulation when investigating VANETs, leaving the hardware side of the research lacking.

1.3 Literature synopsis

Individual subsystems of the proposed system are already in existence. Automatic License-plate Recognition (ALPR) systems using imaging hardware fixed to infrastructure such as lamp poles, bridges and walls are currently in use by government agencies and other organisations to realise applications such as law enforcement, electronic toll collection, average speed detection, traffic control and enterprise security and services [11][12][13]. Mobile ALPR systems mounted on vehicles are also actively in use by law enforcement agencies and some private companies for vehicle identification and data logging [14][15].

MVTRAC is a private company making use of a combination of fixed and mobile ALPR systems to aggregate licence plate recognition data and provide solutions to insurance, finance and government organisations aiding the recovery and repossession of stolen vehicles [14]. Specifics regarding the data aggregation and network implementation is however not disclosed.

Enforcement Deputy Mobile ALPR is a vehicle-mounted ALPR system developed by Roadmetric [15]. It is comprised of cameras mounted in the front, rear, side and optionally the roof of a vehicle with a processing unit running ALPR software. Licence plates can be captured at relative speeds of 240 km/h at distances of up to 25 m, with the processing unit being able to determine the speed of and distance from a detected vehicle. Users of the system are alerted on an LCD monitor when a vehicle of interest, such as a stolen or unlicensed vehicle, is identified by comparing the detection data to a database of known vehicles of interest. ALPR and geographical data obtained form a Global Positioning System (GPS) device is wirelessly uploaded to back-end infrastructure, however a network description is omitted. It is thus not known which wireless technologies and configuration are used.

The proposed system has a similar high-level functionality as the Enforcement Deputy Mobile ALPR system, however the method in which the data is aggregated would differ. Thus, focus will be placed on the networking and communications aspect of the system, since the ALPR aspect of the solution has a long history of widespread implementation and usage in the field.

Data privacy and security are major challenges to overcome in the deployment such systems in public spaces, since geographical meta-data linked to an identity is highly sensitive and can easily be used for malicious intent when fallen into the wrong hands. Implementation of such an unsecured system will raise concerns in the public, and will also be nearly impossible to deploy in

European Union (EU) countries implementing the General Data Protection Regulation. As an example, the United States (US) Department of Homeland Security proposed the implementation of a national licence plate tracking system in 2014, the goal being to aid ongoing criminal investigations, but the plan was cancelled due to privacy concerns raised by the public [16].

Privacy, ethical and security concerns raised by implementation of such a system are known, but will be overlooked in this project since it is not a product development.

VANET has become a popular research topic in recent years. Although much research has been done on VANETs, the technologies have not been widely implemented as of yet. Due to the extent of interest shown by researchers, governments and the car manufacturing industry, many applications for VANETs have been envisioned. The following summarises the main application categories [17]:

- Safety related applications
- Traffic information and management systems
- Transport efficiency
- Infotainment

Safety-related applications are popular due to the great benefit they could provide to driving safety and quality. For example, vehicles can exchange information about themselves and their immediate surroundings to other nearby vehicles and road side units (RSUs), aiming to detect, avoid or anticipate potentially harmful situations [18]. Traffic information and management applications' target areas include toll collection, traffic congestion notification and control, as well as intersection management. Infotainment applications in VANETs are envisaged to provide services like media streaming, internet access, updated map information, Point of Interest (POI) detection, direct marketing, etc [17].

Due to the unique nature and characteristics of VANETs and the envisaged applications, new communication standards have been created specifically for use in these scenarios. IEEE 1609 WAVE (Wireless Access in Vehicular Environments) standard is used in the U.S.[19], ETSI ITS G5 standard is used in Europe [20] and ARIB STD-T109 is used in Japan [21]. These standards are based on an amendment of IEEE 802.11, called IEEE 802.11p, specifically designed to realise wireless communication in vehicular environments. The 5.9 GHz band is used to realise communication in all the abovementioned standards, except for ARIB STD-T109 which uses the 700 MHz band.

In 1999, US Federal Communications Commission (FCC) allocated the 5.850-5.925 GHz band to Dedicated Short-Range Communications (DSRC) [22], and the 5.860-5.9 GHz band is reserved in Europe [23]. These standards and specifications are discussed more detail in Chapter 2.

Routing protocols are responsible for realising the communication path in networks, and dictate how data is communicated between source and destination nodes. This makes routing protocols one of the key components of a successful VANET, especially when multi-hop data dissemination is required. Data needs to be transferred reliably, on-time, and at fairly high speeds while taking into account that the network topology is rapidly changing. This makes it an exciting field of research, resulting in copious amounts of protocols available for VANETs and MANETs. Various types of routing protocols usable in a VANET environment and their characteristics are elaborated on in Chapter 2.

1.4 Problem Statement

To determine which existing VANET standards and protocols are suitable for a distributed license plate detection and reporting system in an urban scenario.

1.5 Objectives of this study

Existing VANET standards and protocols will be selected with which to realise the networking component of the envisaged solution. The intent is not to create new networking solutions or to do an in-depth analysis of all possible networking options that could be used to create such a system, but rather to implement and evaluate selected choices for use in a decentralised licence plate detection and reporting system.

Large scale deployment of commercial or custom-made hardware with which to evaluate the envisaged solution is not possible in this project. A simulation approach will be used instead, with network and traffic simulation executed in parallel. Simulation tools will be selected with which to create a simulation environment capable of simulating wireless communication and vehicular mobility. Certain VANET standards and protocols will be selected with which to create the envisaged solution in in the simulation environment. This simulation environment will be set up in a way that it provides the best possible representation of the real-world scenario achievable with the existing models and allotted time period. Standards and protocols selected for use in simulation will be implemented in hardware on a small scale in order to validate core aspects of the networking simulation, thereby adding trust and validity to

results obtained from the simulation environment. Large scale simulations will then be used to investigate, characterise and evaluate the envisaged solution.

The main objectives of this study can thus be broken down as follows:

- Realise a simulation environment with realistic wireless networking and vehicular traffic components with which to evaluate VANET standards and protocols for suitability of usage in the envisaged solution
- Verify and calibrate the simulation environment by means of a hardware solution
- Execute large-scale simulations with the calibrated and verified simulation environment to extrapolate what the performance of the envisaged solution would be within the scope of the specified scenario, and to determine suitability of the chosen networking standards and protocols for this specific scenario

In addition, the scope applied to the scenario is as follows:

- Research will not be performed on the actual detection of vehicles. This includes optical hardware, image processing, licence plate detection algorithms, etc. It is also not necessary to explicitly implement these features in the simulation
- Existing VANET standards and ad hoc routing protocols will be used (new solutions will not be created)
- Existing network simulation models will be used (no new models will be created from scratch)
- Adhering to specific standards and spectrum usage is not required, since regulations for ITS applications in South Africa has not been finalised
- Physical area of interest is limited to an urban area

1.6 Contributions

During the course of this project, the following contributions to future research were made:

- A simulation environment consisting of both network and traffic simulation was realised. This simulation environment implemented IEEE 802.11p, Ad-hoc On-Demand Distance Vector (AODV) routing protocol, signal propagation effects as well as physical obstructions of correct relative positioning and size in relation to the road network map on which traffic is simulated.

- An affordable, flexible, open-source hardware solution was created which fully implements IEEE 802.11p on the physical as well as the software side. The process of setting up and configuring the implementation was explained to enable future researchers to use this approach as a starting point for their research.

  AODV-UU was updated to support a more recent Linux kernel version, and was successfully implemented as part of the hardware solution. The source code was made available to the public and it is free to use or modify.

  The completed solution can be used as a functional VANET On-board Unit (OBU), and can be implemented by other researchers in the field that are in need of an affordable, open-source VANET hardware solution.

- Empirical data obtained by performing specific experiments with the hardware solution was used to verify and calibrate the core aspects of the networking simulation on a small scale.

- The verified and calibrated simulation environment was used to investigate the performance of a decentralised licence plate detection application realised with IEEE 802.11p and AODV.

1.7 Overview of this work

An overview highlighting the focal aspects of this project is given in this section on a per-chapter basis. This document comprises six chapters. Chapter 1 (this chapter) provides the introduction, setting and overview of the project, while Chapter 2 contains an overview of literature and important aspects that contributes to the understanding and decision making throughout the project. Chapter 3 covers the realisation of the simulation environment which is needed to investigate the selected VANET standards and protocols as they apply to the envisaged solution. In Chapter 4 the set up and creation of a hardware solution which was used to verify and calibrate the simulation environment is outlined. Verification and calibration is also covered in Chapter 4. In Chapter 5 the calibrated simulation environment is used to evaluate the chosen VANET standards and protocols for suitability of usage to realise the envisaged solution on a large scale. Chapter 6 provides a conclusion, recommendations and improvements.

Chapter 2

Literature Study

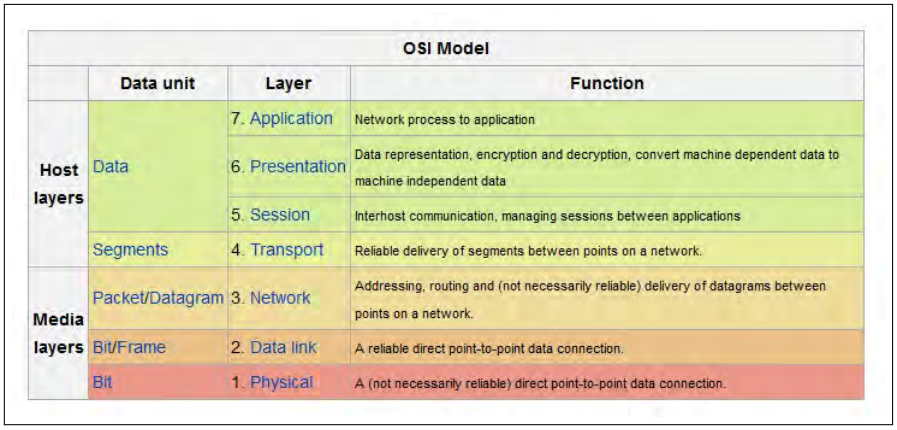
The scope of this project requires a thorough understanding of the technologies used in VANETs and the simulation tools required. Important background knowledge and key aspects are discussed throughout the chapter to provide context and underpin reasoning in upcoming chapters.

This chapter provides an overview of the main networking standards, routing protocols encountered in VANET environments, and commercially available VANET communication hardware. Custom VANET hardware solutions are investigated by reviewing literature in lieu of creating a similar solution in this project. Comparison of various networking and traffic simulation tools and their requirements are discussed using brief overviews.

2.1 Wireless Standards and Protocols in VANETs

The Open Systems Interconnection (OSI) model provides a conceptual seven-layer framework for standard networking protocols, with the goal to abstract each layer from the next and provide interoperability between diverse networking systems. The protocols operating at each layer have a specific function, and only interacts with its adjacent layers. Thus, each layer serves the layer above it and gets served by the layer below it. Figure 2.1 describes each layer and its intended functionality.

IEEE 802.11 standard provides specifications for the Physical (PHY) and Data Link layers (specifically the Media Access Control (MAC) sublayer) in Wireless Local Area Networks (WLAN), and has become the go-to standard for most modern day wireless communication implementations. Adaptations made to this existing standard to suit the conditions in which vehicular networks operate, are the main enabling technology for VANETs.



OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation, encryption and decryption, convert machine dependent data to machine independent data
		5. Session	Interhost communication, managing sessions between applications
	Segments	4. Transport	Reliable delivery of segments between points on a network.
Media layers	Packet/Datagram	3. Network	Addressing, routing and (not necessarily reliable) delivery of datagrams between points on a network.
	Bit/Frame	2. Data link	A reliable direct point-to-point data connection.
	Bit	1. Physical	A (not necessarily reliable) direct point-to-point data connection.

Figure 2.1: Reference OSI Model layers [1]

The networking layer, but more specifically routing protocols, is another key consideration when it comes to vehicular networks. Many of the routing protocols used in VANETs have been brought over from the MANET environment, but can prove ineffective due to the unique characteristics of VANETs. Routing protocols have been a major subject of VANET research due to its significance and effect on the entire VANET architecture. Recently, there have been research papers investigating and proposing routing protocols that are specifically tailored to the VANET environment.

2.1.1 Wireless Access Technologies

This section provides a detailed overview of the main wireless access technologies considered in VANET research. It is necessary to gain a good understanding of the standards and protocols, in order to make appropriate choices regarding the standards to be used throughout this project. The choice of simulation software, physical hardware, making assumptions, etc. will all be influenced by the information in this section, since they form the base of any VANET implementation.

The U.S., Europe and Japan have set forth standards to be used in ITS and VANET applications, all employing different protocols on the PHY and MAC layers. Respectively, these standards are: IEEE 1609 WAVE [19], ETSI ITS G5 [20] and ARIB STD-T109 [21]. However, all three standards are based on an amendment of IEEE 802.11, called IEEE 802.11p, specifically designed to realise wireless communication in vehicular environments.

2.1.1.1 IEEE 802.11p

The IEEE 802.11p standard [7] is an amendment of the IEEE 802.11 standard to support Wireless Access in Vehicular Environments (WAVE), specifically

V2V and V2I communication. It provides specifications for the PHY and MAC layers. IEEE 802.11p inherits the Orthogonal Frequency-division Multiplexing (OFDM) physical layer from IEEE 802.11a, with the main differences being band of operation (5.9 GHz), signal bandwidth and guard interval. Due to timings being doubled compared to 802.11a, the bandwidth is reduced from 20 MHz to 10 MHz and the guard interval is twice that of 802.11a. This theoretically makes signals less prone to fading and increases multipath propagation effect tolerance [24]. Since 802.11a was developed for a relatively stationary environment, the harsh signal propagation environment of VANETs might cause 802.11p to suffer in terms of reliability. As stated in [24], 802.11p's pilot sub-carrier spacing and sub-optimal channel estimation can contribute to degraded performance. Residual frequency offset correction is performed by using only four pilot sub-carriers, and due to the spacing of the sub-carriers not being close enough, accurate sampling of the frequency variation of the channel can become challenging. Channel estimation is performed by placing short and long training symbols in the preamble of each transmitted packet, used for coarse and fine synchronisation respectively. The high time-variance of the channel and unrestricted packet length, combined with the single estimation per packet, can cause unreliable (outdated) packet estimation. One can deduce that minimizing payload packet size might be beneficial in a highly mobile environment.

IEEE 802.11p introduces a mode of operation called 'Outside the Context of a BSS (OCB)' mode. This allows a Station (STA) to transmit data frames while not being a member of a Basic Service Set (BSS). Instead of going through a lengthy procedure to join a BSS, the STA will use well-known values for parameters like modulation and coding scheme.

On the MAC layer, IEEE 801.11p employs the same Hybrid Coordination Function (HCF) channel access method as IEEE 802.11e [8]. HCF combines functions from the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF), with additional, enhanced Quality of Service (QoS)-specific mechanisms. As stated in [8], "The MAC architecture can be described as providing the PCF and HCF through the services of the DCF". For contention-based channel access, HCF uses the Enhanced Distributed Channel Access (EDCA) mechanism, and for contention-free access the HCF Controlled Channel Access (HCCA) mechanism is used. The contention-based EDCA mechanism is designed for prioritised QoS support and defines four Access Category (AC)s: Background (BK), Best Effort (BE), Video (VI) and Voice (VO). Each AC has its own, independent queue and standard channel access parameters, which include the contention window's (CW) minimum and maximum value (aCWmin and aCWmax), Arbitration Inter-frame Space Number (AIFSN) and Transmission Opportunity (TXOP) limit. The AC parameters of IEEE 802.11p and IEEE 802.11e can be seen

in Table 2.1, whilst other related IEEE 802.11p parameters are presented in Table 2.2.

Table 2.1: Default AC parameters of IEEE 802.11p and IEEE 802.11e [7][8]

AC	CWmin	CWmax	AIFSN		TXOP limit		
			802.11p	802.11e	802.11p	802.11e Clause 15 & 18	802.11e Clause 17 & 19
AC_BK	aCWmin	aCWmax	9	7	0	0	0
AC_BE	aCWmin	aCWmax	6	3	0	0	0
AC_VI	(aCWmin+1)/2-1	aCWmin	3	2	0	6.016 ms	3.008 ms
AC_VO	(aCWmin+1)/4-1	(aCWmin+1)/2-1	2	2	0	3.264 ms	1.504 ms

Table 2.2: Default IEEE 802.11p parameters [7]

Parameter	Value
aCWmin	15
aCWmax	1023
aSlotTime	13 μs
aSIFSTime	32 μs

The working of the EDCA mechanism can shortly be described as follows: If a frame arrives at an AC's queue, the station (STA) will sense the channel to check if it is idle or busy. If the channel is sensed to be idle, and the queue has no backlogged data, a transmission will be initiated if the channel stays idle for AIFSTime. AIFSTime is calculated as

$$AIFSTime[AC] = AIFSN[AC] \times aSlotTime + aSIFSTime \tag{2.1}$$

where aSIFSTime is the duration of the Short Inter-frame Space (SIFS) and aSlotTime is the slot duration. If the channel is sensed to be busy, the STA will continue to sense the channel until it becomes idle. The backoff procedure will be initiated if the channel remains idle for AIFSTime, with the backoff counter being set to a random value selected from the range [0, CW], where CW = aCWmin. The counter will be decremented by 1 each time the channel is sensed to be idle in a slot. If the channel is sensed to be busy during the backoff period, the timer will be frozen until the channel is sensed to be idle for a period of AIFSTime, after which it will continue the decrement process. A transmission will be initiated once the backoff timer reaches zero. If the transmission attempt fails due to the lack of frame Acknowledgement (ACK), a new backoff procedure will be initiated. After each retransmission attempt the value of CW is doubled until it reaches aCWmax, after which it will stay at aCWmax. CW will be reset to aCWmin after a successful transmission, or if the retransmission limit is reached. In IEEE 802.11e, TXOP is granted to an AC when it determines that it can initiate a transmission. TXOP is a

time interval (limited to TXOP limit) in which an AC can initiate multiple frame-exchange sequences, separated by SIFS.

An external collision can occur when different STAs grants TXOP to one of their ACs, since there are no priorities among STAs. Thus, all STAs must compete for access of the same medium with equal priority. However, IEEE 802.11p sets the TXOP for each AC to zero. A value of zero means that, when the AC is granted TXOP, it can transmit a single MAC service data unit (MDSU) regardless of its length. An internal collision can occur if multiple AC queues in a STA wants to initiate a transmission at the same time. Access will be granted to the AC queue with the highest priority, whilst the other AC queues will initiate the backoff procedure as if it was an external collision (however the retransmission counter is not increased).

2.1.1.2 IEEE 1609 WAVE Family of Standards

The IEEE 1609 family of standards [19] was developed specifically for application in VANETs. Defined by IEEE 1609 is an architecture and a complementary set of interfaces and services, which collectively aim to provide secure V2V and V2I communications. It builds upon the IEEE 802.11p standard which defines the PHY and MAC layers, whilst extending the MAC layer and providing support for networking services, security services and resource management. Figure 2.2 provides an overview of the IEEE 1609 WAVE architecture and how it builds upon IEEE 802.11p. IEEE 1609.1 defines the resource manager (RM) and its interfaces and services, IEEE 1609.2 specifies security services for management messages and applications, IEEE 1609.3 specifies networking services and provides the WAVE Short Message Protocol (WSMP) and IEEE 1609.4 provides support for multichannel operation [19]. Additional management functions are also implemented for the PHY and MAC layers, named PLME and MLME.

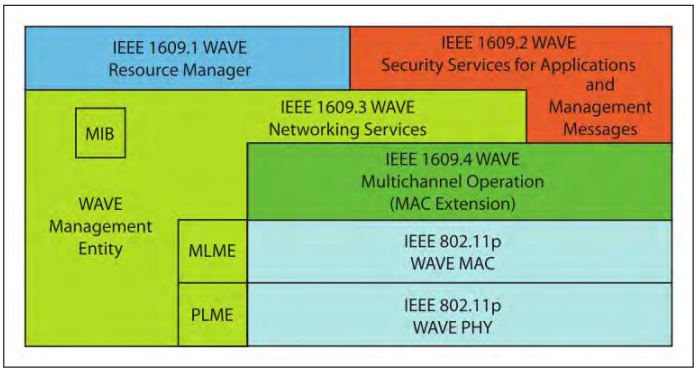


Figure 2.2: IEEE 1609 WAVE architecture [2]

IEEE 1609.1 [25] specifies the RM application which allows remote applications to communicate with On-Board Units (OBUs) through Road Side Units (RSUs). It behaves like an application layer, with the purpose of multiplexing communications from various remote applications, whilst each remote application communicates with multiple individual OBUs. The main goal is to provide support for a wide range of applications for OBUs by enabling interoperability of applications using WAVE. The standard defines specific application-independent commands and message formats to be used by applications. This means that when an application uses RM, it does not need to run on the OBU itself. The application could run in remote locations or RSUs; thus the processing, management and encryption workloads does not need to be the responsibility of OBUs. This can potentially reduce OBU cost and increase performance.

IEEE 1609.2 [26] specifies security services for WAVE Short Messages, WAVE Service advertisements and additional security services that may be used by upper layers in the stack. These services are categorised as management functions and processing functions, where management functions are responsible for certificate maintenance and key generation, and processing functions are responsible for securing data and communications. For digital signatures, the Elliptic Curve Digital Signature Algorithm (ECDSA) is used over P-244 and P-256 NIST elliptic curves, with SHA-244 and SHA-256 hashing algorithms respectively. As for encryption, both asymmetric and symmetric algorithms are supported. For asymmetric encryption, the Elliptic Curve Integrated Encryption Scheme (ECIES) over the P-256 NIST elliptic curve is used. Advanced Encryption Standard (AES) in Counter Mode with Cipher Block Chaining Message Authentication Code (CCM) is the supported symmetric encryption algorithm.

IEEE 1609.3 [27] specifies the networking services operating at the network and transport layers, which provide routing and addressing to support high priority, secure communication. This standard introduces WAVE Service Advertisements (WSA), channel scheduling and the WAVE Short Message Protocol (WSMP). Networking functions specified by IEEE 1609.2 can be divided into two types, namely data-plane services and management-plane services. In the data-plane, the WSMP as well as Internet Protocol version six (IPv6) stacks are supported, which both operate on a single Logical Link Control (LLC) layer. WSMP is intended to be used for high priority communications (usually single-hop, safety related data dissemination to nearby nodes), whilst non-safety relayed data transfer is handled by Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). Management-plane services include WAVE Basic Service Set (WBSS) management, IPv6 configuration, channel usage monitoring, application registration and Management Information Base (MIB) maintenance.

IEEE 1609.4 [28] specifies the MAC sublayer services and functions to support multi-channel communication support. As mentioned earlier, IEEE 1609.4 acts as an extension of the MAC layer defined by IEEE 802.11p. It defines seven channels - one control channel (CCH) and six service channels (SCH), each with specific frequencies and characteristics. Each channel is designed with specific applications in mind. Figure 2.3 indicates each channel's characteristics and intended application use cases. A synchronised scheme based on Coordinated Universal Time (UTC) is used for channel coordination. This ensures that all nodes connected to a WBSS are synchronised and capable of monitoring the CCH and SCH during a common time interval. The CCH is served every other timeslot, whilst all the SCHs utilise the remaining time slots. Individual SCHs do not have a specific slot assigned to them, the timeslots not used by the CCH are assigned to SCHs depending on application requirements. Unlike IEEE 802.11p that only supports single channel operation, the CCH and SCHs in IEEE 1609.4 have unique EDCA AC parameters. Only certain types of frames may be transmitted in certain channels. CCH only allows WSA and WSM frame transmission, while other IEEE 802.11 management frames and IP data frames may utilise the different SCHs. Nodes, being members of a WBSS, is a prerequisite of SCH frame exchange initiation.

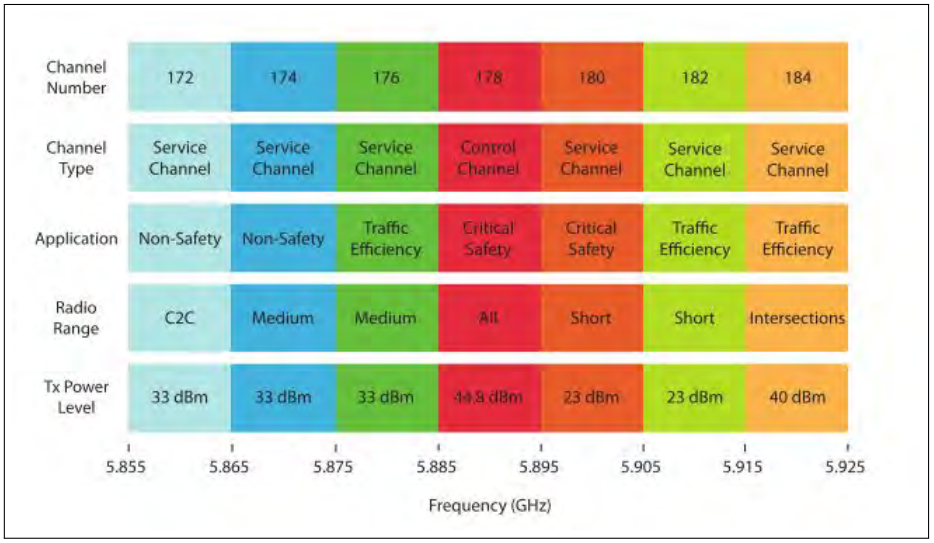


Figure 2.3: IEEE 1609 WAVE channel allocation [2]

2.1.1.3 ETSI ITS Standards

In Europe, the European Telecommunications Standards Institute (ETSI) developed standards to support intelligent transportation systems (ITS), of which VANETs are just a piece of the puzzle. Intelligent Transport Systems (ITS);

Communications Architecture by ETSI [20] specifies an open systems communication architecture for ITS, based on ETSI ITS and IEEE standards. ETSI TR 101 607 [29] provides a breakdown of all the applicable standards and specifications to be used in the aforementioned architecture. A brief summary of the ETSI ITS architecture is depicted in Figure 2.4.

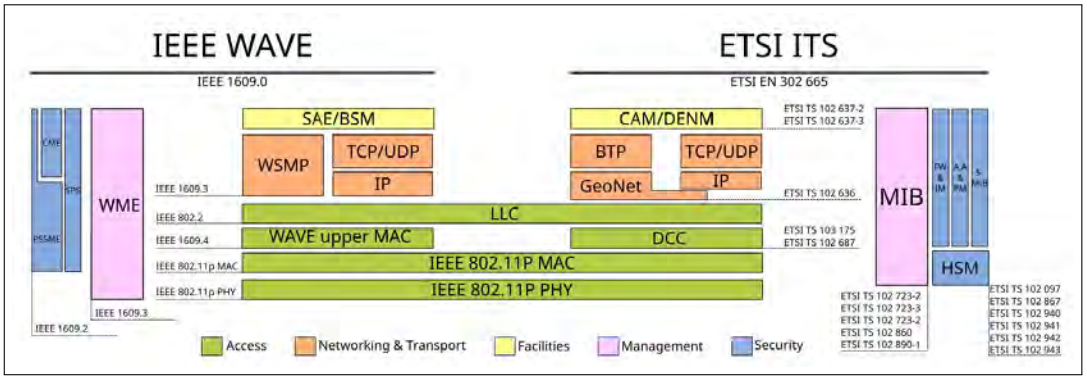


Figure 2.4: ETSI ITS architecture [3]

ITS-G5. Standards that specify the physical and data link layers of the protocol stack are collectively called ITS-G5. As seen in Figure 2.4, IEEE 802.11p is used for the PHY and MAC layers. Unlike IEEE 1609, stations implementing the ITS-G5 stack can operate in four different frequency bands, namely ITS-G5A, ITS-G5B, ITS-G5C and ITS-G5D. Each frequency band is dedicated to specific applications, as shown in Table 2.3. Moreover, ITS-G5 has eight channels of operation - one control channel (G5-CCH) and seven service channels (G5-SCHs). All channels have a fixed center frequency (except G5-SCH7) and falls within a specific 10 MHz frequency band, as indicated by Table 2.3. As in IEEE 1609, the CCH may only be used for safety-critical messages. However, in ITS-G5, dedicated transceivers are required for control and service channels, which eliminates the risk of losing safety-critical packets due to channel switching. A MAC extension layer called Decentralised Congestion Control (DCC) [30] is added to the data link layer in order to control channel load by dynamically changing channel access rules. This is achieved by the implementation of a state machine with three states: relaxed, restrictive and active. Each state has unique parameters for transmission power, receiver sensitivity, PHY layer data rate and interval between packets. State transitions are based on the minimum and maximum channel load observed within a certain period - 1 second and 5 seconds respectively.

Network and Transport layers. ETSI TS 102 636 specifies two new protocols - GeoNetworking [31] and Basic Transport Protocol (BTP) [32]. GeoNetworking is an ad hoc network layer protocol that supports multi-hop data

Table 2.3: ETSI ITS channel specification [9]

Band	Channel	Frequency Rage [MHz]	TX Power Density Limit	Usage	Standard
ITS-G5A	G5-CCH	5 895 to 5 905	23 dBm/MHz	Safety Related Applications	EN 302 571
	G5-SCH1	5 875 to 5 885	23 dBm/MHz		
	G5-SCH2	5 885 to 5 895	13 dBm/MHz		
ITS-G5B	G5-SCH3	5 865 to 5 875	13 dBm/MHz	Non-Safety Related Applications	
	G5-SCH4	5 855 to 5 865	-10 dBm/MHz		
ITS-G5D	G5-SCH5	5 905 to 5 915 5	-10 dBm/MHz	Future Applications	
	G5-SCH6	5 915 to 5 925	-10 dBm/MHz		
ITS-G5C	G5-SCH7	5 470 to 5 725	17 dBm/MHz (DFS master) 10 dBm/MHz (DFS slave)	RLAN (BRAN, WLAN)	EN 301 893

dissemination by using the geographical location of nodes. Unlike addressing in IPv4 or IPv6 that uses a IP address to identify nodes, GeoNetworking uses geographical addressing that enable nodes to send data to a specific geographical region or node at a specific position. Each packet contains the geographical location of the destination, which nodes use to make forwarding decisions based on the current network topology. This does however require each node to have at least partial information regarding the topology of the surrounding network. GeoNetworking also supports transparent routing of IPv6 packets, which allows an IPv6 application to run on top of the GeoNetworking protocol. BTP is similar to UDP in the sense that they both provide contention-less communication with minimal overhead, no guarantee of delivery or ordering, and no duplicate protection. It relies on the lower layers to provide reliable packet delivery and assumes that nodes using the protocol are aware of the potentially unreliable data transfer. BTP is designed to multiplex and demultiplex data from entities at the Facilities layer (described in the next part).

The Facilities layer, as defined by TS 102 637 [33], covers the session and presentation layers. Facilities are services running in the Facilities layer that provide services and functions for the ITS Basic Set of Applications (BSA), as well as ensuring interoperability between applications. Core services like time management and Cooperative Awareness Message (CAM) management are provided by common facilities, whilst application specific services like the decentralized environmental notification (DEN) management are provided by domain facilities. The CAM management service [34] periodically generates CAM messages that are disseminated to nearby nodes. CAMs contain information about a node's current position, movement, sensor information and other relevant attributes. CAMs are broadcasted periodically at rates between 1 and 10 messages per second to neighbours that are within a single hop. The broadcast interval is adjusted according to the importance of a message, e.g. an intersection collision warning will be broadcasted 10 times per second, whilst a speed limit notification would be broadcasted once per second. The DEN management service [35] generates and processes Decentralized Environmental Notification Messages (DENM). DENMs can contain information regarding

the current driving environment or detected traffic events, and are also broadcasted periodically. The main difference between CAMs and DENMs is that DENMs are broadcasted to nodes within a specific geographic location, rather than solely to nearby neighbours. The Cooperative Road Hazard Warning (RHW) application is the main user of DENMs, with the creation of DENMs being triggered by events like the detection of a traffic jam or low visibility conditions.

Management and Security layers are specified by the standards indicated in Figure 2.4, and are not covered in detail in this report.

2.1.1.4 ARIB STD-T109

In Japan, the Association of Radio Industries and Businesses (ARIB) created the STD-T109 standard [21] for ITS application in the 700 MHz band. Its main goal is to provide safety-related information to drivers via low latency V2V and V2I communications. The standard specifies four layers based on the OSI model - physical layer, data link layer, the Inter-Vehicle and Roadside-to-Vehicle Communication (IVC-RVC) layer which implements functionality generally found in layers 3 to 6, and "Layer 7". Figure 2.5 provides an overview of the ARIB STD-T109 protocol stack.

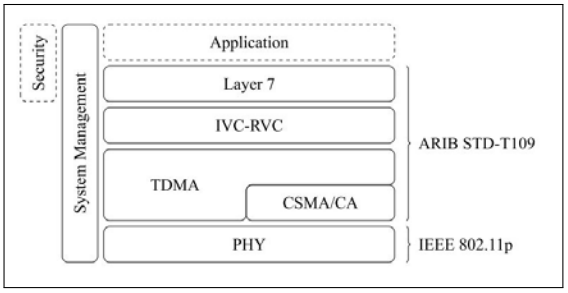


Figure 2.5: ARIB STD-T109 architecture [4]

The PHY layer implements IEEE 802.11p's physical layer, but operates at a 760 MHz center frequency. Unlike the previously discussed ITS standards, STD-T109 makes a clear distinction between V2V and V2I communication and does not implement channel switching since it only operates in a single channel. A unique medium access scheme is used - a combination of carrier-sense multiple access with collision avoidance (CSMA/CA) and a virtual carrier sense function that utilises time-division multiple access (TDMA) is implemented.

The MAC sublayer manages the 1 second cycle timer that controls V2V and V2I communication. It also manages the transmission inhibition period, a period during which a station is not allowed to transmit frames. Information needed to set the transmission inhibition periods is generated and maintained

by the IVC-RVC layer. The 1 second cycle timer is divided up into ten communication cycles of 100 ms each, during which RSUs and mobile nodes perform communications. These cycles are yet again divided up into smaller cycles of 6.24 ms each, consisting of 16 sub-cycles in each 100 ms cycle. This 6.24 ms cycle is further split into two periods - one dedicated to RSUs (RVC period), while the remaining period may be used by mobile nodes to compete for channel access. The RVC periods would thus be the transmission inhibition periods for mobile nodes. A dedicated transmission period within the RVC period may be assigned to a specific RSU, in which only it may access the channel. This period would thus be the transmission inhibition period for other RSUs in the surrounding area. CSMA/CA is not needed in this case, since there is no need to compete for channel access. Mobile nodes are informed of RVC periods during each of the 6.24 ms cycles. RSU's disseminate this information via multi-hop broadcast messages limited to a certain number of hops. Time synchronisation between mobile nodes are performed by inserting the local time at the sender into the frame header, whilst RSUs use external sources to synchronise their clocks. RSUs can also synchronise their clocks with the same process as mobile nodes.

During a RVC period, a RSU may transmit a frame if the medium is sensed to be idle by the virtual carrier sense function. The medium is sensed as idle by the virtual carrier sense function if the transmission timing is not within a transmission inhibition period. However, it must wait for a period called 'shortest space' (32 μs) before initiating the transmission. Frames may be transmitted until the virtual medium is sensed to be busy, with a period of 'shortest space' between frames. Mobile nodes are permitted to transmit a frame if the physical carrier sense function senses the medium to be idle for a period of $ShortestSpace + 2 \times SlotTime$, and the virtual carrier sense function senses the medium to be idle. Before initiating the transmission, the station initialises a counter equal to $RANDOM \times SlotTime$, where RANDOM is a random integer between 0 to 63 selected from an uniform distribution. The counter is decreased by 1 each time the medium is physically sensed to be idle, and the frame may be transmitted once the counter reaches zero. If the medium is sensed to be busy by the physical carrier sense function before the counter reaches zero, the entire process is repeated until the virtual carrier sense function senses the medium to be busy.

Layer 7 aims to provide services and functions to applications, however developing the applications is left to working groups. Layer 7 can communicate with applications, security management entity and system management entity through defined services. Service users must make use of service primitives to access these services.

2.1.2 Spectrum Allocation

Early in the 21st century, governments around the world started to realise the potential of large scale Intelligent Transportation Systems. Regulatory bodies in the United States of America (USA), Europe and Japan allocated frequency spectrum specifically for DSRC. In 1999, the US Federal Communications Commission decided to allocate the 5.850-5.925 GHz band exclusively for DSRC use [22], and adopted DSRC service rules in the 5.9 GHz band in 2003. In Europe, the European Commission (EC) Electronic Commissions Committee (ECC) dedicated the 5.875-5.905 GHz band to safety related ITS applications in 2008 [36]. In Japan, a single 10 MHz channel (755.5-764.5 MHz) has been allocated for ITS use.

In South Africa, no frequency band has yet been allocated specifically for ITS use [37]. This may be a potential issue when developing such systems here. South Africa follows most of the spectrum allocation trends of Europe, increasing the probability of similar future ITS spectrum allocation locally [38].

2.1.3 Routing Protocols

In a distributed environment or network where the source and destination nodes require communication via intermediate nodes, routing protocols are a key component for successful communication. They determine the optimal paths for data to travel between a source and destination node. This process is known as *routing*, and routing protocols make use of routing algorithms to achieve this. Additionally, how information is shared with neighbouring nodes and other nodes throughout the network is dictated by the routing protocol [39]. Essentially, routing protocols specify or standardise how data is disseminated between nodes, implementing the same protocol, and computes the optimal path for data to travel by using a routing algorithm.

Due to the volatile mobility of nodes in a VANET, establishing and maintaining routes for multi-hop data transfer can be very difficult. Such a network can experience rapid changes in topology as well as node density, which can lead to frequent link breakages and the need to establish new routes. As previously stated, a substantial portion of research in the VANET field has been aimed at evaluating, improving and creating routing protocols as the success of a VANET is dependent thereon. Routing protocols in VANETS can be divided into the following categories: Topology-based, cluster-based, position-based, and geocast.

2.1.3.1 Topology-based Protocols

Topology-based routing protocols make use of known link state information in the network to realise packet forwarding. Three sub-types of this category include proactive (table-based), reactive (on-demand) and hybrid routing [40].

Proactive Protocols (table-based) As the name suggests, this type of routing protocols uses tables to store and maintain routing information of all associated nodes in the network. In doing so, routes can be established without any delay or a route discovery process. However, when the topology changes, control messages are broadcast by affected nodes to update their routing tables. This can cause substantial management overhead when topology changes are frequent, which is the case in VANETs. The routing tables will be very large in large networks and significant processing resources can be consumed when a route needs the be established. Due to VANETs being highly mobile, and usually consisting of a large number of nodes, proactive routing protocols may not be the optimal choice for these scenario types. Well-known proactive routing protocols include OLSR, GSRP, and DSDV.

Reactive Protocols In contrast to proactive routing protocols, reactive routing protocols (also known as ad hoc or on-demand protocols) only establishes a route when necessary. When a route needs to be established, the source node initiates the route discovery process by broadcasting control messages to neighbouring nodes. These messages propagate through the entire network until it reaches the destination node. This process is known as flooding, and may induce overhead throughout the network. In an environment where link breakages are frequent, network performance can suffer greatly due to the flooding of control messages. Well known reactive routing protocols include AODV, DSR, and PRAODV [41].

Hybrid Protocols Hybrid topology-based routing protocols are a combination of proactive and reactive protocols. Most commonly, hybrid routing protocols divide nodes into different zones to aid route discovery and maintenance. Usually, proactive routing is used within zones, whilst reactive routing is used outside zones. This results in fast link establishment inside zones as well as less management overhead in the overall network when compared to reactive and proactive routing protocols. However, these protocols were not designed for networks that rapidly and frequently undergo changes in topology. ZRP, HARP, and ZHLS are examples of hybrid routing protocols [40].

2.1.3.2 Position-based Protocols

Position-based routing protocols make use of nodes' current geographical location, obtained via on-board devices like GPS units, to make forwarding de-

cisions. No routing table is maintained, nor is any information regarding link status - nodes merely share their current position with neighbouring nodes via beaconing. Most position-based routing protocols make use of greedy forwarding or improved greedy forwarding. This approach implies that nodes forward their packets to a neighbouring node that is closest to the destination node. Some protocols also make use of geographical map information to assist routing decisions. Position-based routing protocols can struggle in an urban environment, since a node that is geographically the best option for forwarding a packet to, may be obstructed by obstacles like buildings or trees. Popular position-based routing protocols include: GPSR, GSR, and A-STAR.

2.1.3.3 Cluster-based Protocols

Cluster-based routing protocols create virtual network infrastructure by clustering together nodes that are near each other. Each cluster has a 'cluster head' node which is responsible for inter-cluster communication and intra-cluster coordination, whilst intra-cluster communication is performed by creating direct links between nodes in a cluster. Since the cluster head has a lot of inherent responsibility, choosing the correct node is essential. Due to the dynamic nature of VANETs, creating clusters, managing clusters and choosing suitable cluster heads can be challenging. Current clustering techniques used in MANETs are unstable when applied to VANETs [41], due to the aforementioned reasons. However, some researchers have proposed clustering techniques for application in VANETs, namely Clustering for Open IVC Networks (COIN) and LORA-CBF [42].

2.1.3.4 Geocast Protocols

Simply put, geocast routing is essentially a multicast service within a fixed topographical area or zone of relevance (ZOR) [40]. The goal of this implementation is to disseminate packets from the source node to all the nodes in the ZOR. A forwarding zone is defined where the flooding of packets is directed to stay within the zone, thus avoiding congestion due to flooding outside the ZOR. Within the destination zone, packets can be forwarded with unicast routing. AGR and IVG are examples of geocast routing protocols [41] [40].

2.1.4 Summary

This section provided an overview of the different VANET wireless access technologies, ITS spectrum allocation in South Africa, and common routing protocol types.

South Africa does not have spectrum allocation for ITS applications, and it was assumed that South Africa will follow the spectrum allocation trends of

Europe as mentioned in Section 2.1.2. Frequencies utilised by ETSI ITS-G5 and IEEE 1609 WAVE fall within the spectrum allocation reserved for ITS applications, ruling out AIRB STD-T109 as an usable standard in this project since it operates in the 700 MHz band.

It was stated that the envisaged solution could be a standalone application that will not be integrated with existing ITS infrastructure and systems. Thus it is not required for the solution to specifically implement either ITS-G5 or IEEE 1609. It would be possible to create the envisaged solution solely with IEEE 802.11p without relying on the upper layer network specifications provided by the aforementioned network standards. IPv4/IPv6 and TCP/UDP can be used in conjunction with an application upon the PHY and MAC layers provided by IEEE 802.11p.

Based on this and and the assumption that it would be difficult to obtain comprehensive, fully working, open-source implementations of ITS-G5 or IEEE 1609 WAVE, it follows that IEEE 802.11p can be chosen as the network standard for this project.

Most VANET applications are safety-related or traffic information related, which implies that the nodes are actively interested in the data received from neighbours and other nodes in the network. However, the application envisaged for this project and characteristics thereof, is different from the conventional VANET applications: Nodes will only act as data generators and forwarders, with no self-interest in the generated data. All data has a single destination, which implies multi-hop data transfer given the scenario. Low network latency is not required since the data is not as time-sensitive as in the case of safety-related applications. As described in Chapter 1, the scope of the project is limited to an urban scenario which only considers limited or no LOS between source and destination, and frequent link breakages due to obstructions. Vehicles will however travel at lower speeds in an urban area with the possibility of frequent stops, which could allow for larger windows of opportunity to establish and maintain links in these situations.

Given these characteristics of the envisaged application, certain types of routing protocols would not be suitable. Nodes will merely act as data generators and data forwarders to propagate data to the nearest fixed aggregator, therefore geocast and broadcast-based routing protocols do not seem optimal in this case. Section 2.1.3.4 states that geocast routing is effective when there are multiple destination nodes in a specific zone (ZOR), and as mentioned earlier in this section, broadcast-based routing would be most effective for safety- or traffic-related applications since data is disseminated to all available nodes within a certain amount of hops or distance.

Cluster-based routing protocols create virtual network infrastructure by grouping nodes that possess similar characteristics such as position and velocity. As

stated in Section 2.1.3.3, a cluster head is needed to manage intra-cluster and inter-cluster communication, hence the importance of choosing the correct node to act as the cluster head. In a highway scenario this could be an easier task than in urban environments, due to the infrequent changes in vehicle speed and direction. It can be seen that cluster-based routing protocols may suffer from severe instability issues in an urban environment, thus eliminating it as a possible option for this project.

Position-based routing protocols make use of accurate map information along with a node's current geographical location. Since vehicles in the envisaged scenario will be equipped with GPS units, this protocol category cannot be ruled out yet. However, Section 2.1.3.2 explains that in an urban environment, attempting to establish a link based on geographical position may not be the optimal solution due to various possible obstructions. Due to vehicles rarely remaining stationary in a VANET, the chance of nodes having up-to-date information regarding neighbouring nodes' positional information is also fairly slim. Positional-based routing is thus also ruled out as a possible solution in this project.

Given the characteristics of this specific scenario and application, topology-based routing protocols seem more suitable than the various protocol categories discussed above. Routing protocols in this category make use of link state information in the network to either pro-actively or reactively make routing decisions, as mentioned in Section 2.1.3.1. It is, however, not a perfect solution, since frequent topology changes and link breakages will still cause a degree of flooding.

2.2 VANET Simulation Environments

VANET simulations require two main aspects - simulation of vehicle mobility and simulation of the wireless network. The network simulation relies on the vehicle mobility simulation to provide positional information of nodes, in some cases, the reverse is required. The mobility simulation requires the network simulator to provide information to update the vehicle movement based on network events. For example, when a traffic jam is detected, vehicles in the network will be notified of this event to choose a more optimal path of travel. Generally, network simulators and vehicle mobility/traffic simulators are independent tools that need to communicate via one-way or two-way interfaces depending on the use case. There are solutions that offer both network and vehicle mobility simulations as part of a single package, but these tools do not have the extensive features and customisation options provided by the well-established, independent tools.

This section will provide a brief overview of the available network and mobility simulation tools and compare their features.

2.2.1 Vehicle Mobility / Traffic Simulators

Realistic vehicle mobility is a key aspect of VANET simulations. Most of these simulators were developed to aid road planning or to analyse existing road infrastructure, and were not developed with VANETs in mind.

Accurately simulating traffic is a daunting task due to the countless number of variables that can influence a vehicle's behaviour, like traffic rules, interaction with other vehicles and the surrounding environment and variance of individual driver characteristics. Models of vehicle mobility can be classified as macroscopic, microscopic or mesoscopic [43].

Macroscopic models describe traffic as flows with a certain density and velocity, whilst considering constraints like roads, traffic lights and intersections. Microscopic models specify the mobility characteristics of individual vehicles, and their interaction with other vehicles in the simulation. Mesoscopic models combine the characteristics of macro- and microscopic models, but can lack the ability to portray detailed vehicle characteristics [44].

It can be deduced that accurate microscopic models will be the most beneficial option in VANET simulations, due to the refined level of granularity available on a per-vehicle basis. In addition to this, realistic VANET simulations require accurate models of geographical maps that contains roads, infrastructure such as intersections and traffic lights, and obstacles such as buildings. Another key aspect is traffic demand models. These models are created by traffic generators that can take real-world or synthetic data to produce accurate models of traffic densities and flows. This can be done by defining the number of routes (or trips), the start- and endpoint of each route and the path of travel to be taken during a trip [45]. Traffic generators can be included in mobility simulators, but are also available as separate tools.

In summary, it can be said that an ideal vehicular mobility simulator needs to meet the following criteria and requirements to be suitable for realistic VANET simulations:

- Ability to utilise accurate geographical maps
- Ability to create or make use of accurate traffic demand models
- Accurate mobility models that consider elements such as traffic rules, interaction with other vehicles in the simulation and realistic car following models (e.g. Krauss model [46])
- Accurate traffic demand model

Table 2.4: Comparison of vehicular mobility simulators

Features	Simulators					
	SUMO	STRAW	VanetMobiSim	MOVE	CHARISMA	CanuMobiSim
Software Related						
Open-Source	✓	✓	✓	✓	-	✓
Visualisation	✓	✓	✓	✓	✓	✓
Network Simulator Compatibility (Trace Generation)	NS-2, GloMoSim, QualNet	Swans	NS-2, GloMoSim, QualNet	NS-2, GloMoSim, QualNet	NS-2, GloMoSim, QualNet	NS-2, GloMoSim, QualNet
Macro-Mobility						
Custom Maps	✓	-	✓	✓	-	✓
Random Maps	Grid, Spider	×	Voronoi	Grid, Spider	-	-
Multi-lane Maps	✓	×	✓	✓	✓	×
Trip Generation	Random S-D, Activity	Random S-D	Random S-D, Activity	Random S-D, Activity	Random S-D	Random S-D, Activity
Path Computation	RWalk, Dijkstra	RWalk, Dijkstra	RWP, Density, Dijkstra, Speed	RWalk, Dijkstra	Dijkstra, Speed, Density	RWP, Density, Dijkstra
Micro-Mobility						
Following Model	Krauss	Nagel Schreckenberg	×	Krauss	Krauss	×
Lane Changing	×	×	MOBIL	×	×	×
Intersection Management	✓	✓	✓	✓	✓	×
Speed Limits	✓	✓	✓	✓	-	-
Traffic Signs	Stochastic Turns	Traffic Lights	Traffic Lights	Stochastic Turns	Stop Signs	×
Overtaking	-	-	✓	-	-	-

- Interface that can be used to communicate with network simulators

Table 2.4 provides a comparison of various vehicle mobility simulators, based on the work of [43], [47] and [48].

2.2.2 Network Simulators

Network simulators are software tools that model, predict and calculate interactions between network entities. Depending on the specific simulator, models of various protocol stacks are provided, as well as models for effects like path loss, interference and antenna patterns. Due to VANETs being a relatively new field, most network simulators do not provide comprehensive support for all the protocols and architectures mentioned in Section 2.1. For simulation of VANETs, the requirements when choosing a network simulator will vary depending on the scenarios desired to be simulated. However, in general, a network simulator must at least provide support for the following points to be considered as a viable option for VANET simulations:

- Integration with vehicle mobility simulators / provide a built-in vehicle simulator
- Provide models for the IEEE 802.11p standard
- Provide customisability of the various OSI layers / provide an option do create and integrate custom layers

- Support various signal propagation effects
- Support obstacles like buildings
- Scalability with a large number of nodes
- Provide comprehensive result outputs

2.2.2.1 NS-2 and NS-3

NS-2 [49] and NS-3 [50] are popular open-source, customisable, discrete event network simulators for both wired and wireless networks, intended for use within a UNIX environment. NS-2 is outdated (released in 1996) and is being replaced by NS-3 (released in 2008) since active maintenance ceased in 2009.

Counterintuitively, NS-3 is not an upgrade or improvement of NS-2, but is, in fact, an entirely different software written from scratch. As a result, NS-3 delivers increased performance, better scalability, easier memory management, and superior visualisation capabilities when compared to NS-2. NS-2 is also more complex to set up and use than NS-3, especially for newcomers.

NS-2, however, is still a good contender due to its popularity among networking researchers, the vast amount of models and features available that are not yet present in NS-3, as well as support and extended functionality provided by community members. Among the models supported are a wide range of routing protocols, as well as IEEE 802.11p. Both simulators do, however, provide models for each layer of the entire network protocol stack as well as modelling of environmental effects. NS-2 and NS-3 can also incorporate trace files exported from SUMO (albeit with pre-processing), to simulate vehicular traffic within the network simulator.

2.2.2.2 OMNeT++ and Veins Framework

OMNeT++ [51] is generally not seen as a network simulator itself, but primarily as a framework with which to create network simulators. It is a discrete event simulator that can be used to create wired, wireless, on-chip and even queueing networks. OMNeT++ is extremely modular and provides support for various extensions and integrations. The simulator kernel is written in C++, whilst the NED language is used to create and configure simulations. Console-based and GUI-based simulation execution is supported across various operating systems. Veins [52] is an open-source vehicle network simulator that uses OMNeT++ for network simulation, and SUMO for traffic simulation. It provides various models that are of interest in VANET simulations, that include IEEE 802.11p, IEEE 1609, AIRB STD-T109, efficient obstacle shadowing and path loss. Simulations using ETSI ITS-G5 can also be performed with Veins by using the Artery extension. Veins also provides Veins-INET, which enables Veins to be used as the mobility model in the INET framework.

INET [53] is a popular open-source model package that can be used for wired and wireless network simulation in OMNeT++. It contains models for various networking protocol stacks, environmental effects, and node mobility. IEEE 802.11p is supported by the latest version of INET at the time of writing.

2.2.2.3 GloMoSim and QualNet

Global Mobile Information System Simulator (GloMoSim) [54] is a discrete event simulator specifically designed for large scale wireless networks. It was developed with Parsec [55], a simulation language based on C, which enables GloMoSim to execute simulations in parallel and support copious amounts of nodes per simulation. It provides models for various wireless protocols, and enables users to write custom modules. Custom modules must be written in Parsec, since the Parsec compiler is used to compile the simulation. Drawbacks of GloMoSim include: No active development community, complex to use and modify, and lack of proper documentation.

QualNet [56] is a commercial network simulator based on GloMoSim. QualNet has a simple GUI, and is compatible with Linux, macOS and Windows. It can be seen as an easy-to-use version of GloMoSim with an expanded feature set.

2.2.2.4 NetSim

NetSim [57] is a proprietary, easy to use discrete event simulator and emulator for several types of wired and wireless networks. It provides libraries for various network protocols coded in C, as well as the option to modify the libraries. A development environment is provided via an easy to use GUI, with extensive result analysis tools. The built-in emulator can be used to test real applications on the simulated network. Integration with SUMO is provided, but NetSim lacks support for various protocol stacks used in the VANET environment.

2.2.2.5 OPNET

OPNET [58] is another discrete event simulator aimed towards wired and wireless network simulations. It provides users with a GUI to build networks containing each layer of the OSI stack. It uses C and C++, on Windows. It provides libraries for various networking protocols with an object-orientated approach. Interfaces to communicate with external tools is also provided, as well as a built-in traffic simulator.

2.2.2.6 Simulator Comparison

Overviews of other available network simulators can be found in [59] and [60]. Table 2.5 provides a comparison between the features of the discussed network

Table 2.5: Comparison of network simulation tools

Features	**Network Simulators**						
	NS-2	NS-3	OMNeT++		QualNet	NetSim	OPNET
			Inet Framework	Veins Framework			
Software Features							
Integration with Mobility Simulator	✓	✓	✓ (with Veins_Inet)	✓	✓	✓	✓
Operating Systems	Linux, macOS	Linux, macOS	Linux, Windows, macOS		Linux, Windows	Windows	Linux, Windows
Scalability	Medium	Large	Medium		Very Large	Large	Large
Customisable Libraries	✓	✓	✓		×	×	×
Built-in Result Analysis	✓	✓	✓		✓	✓	✓
Free to Use	✓	✓	✓ (For Academic Use)		×	✓ (For Academic Use)	✓ (For Academic Use)
Available Models							
Signal Propagation Effects	✓	✓	✓		✓	✓	✓
Obstacle Models	×	×	✓		✓	×	-
IEEE 802.11p	✓	✓	✓		✓	✓	-
IEEE 1609	×	✓	×	✓	×	✓	-
ETSI ITS	×	×	×	✓	×	×	-
ARIB STD-T109	×	×	×	✓	×	×	-
Routing Protocols	AODV, DSR, DSDV, OLSR, TORA, PUMA, M-DART, community created protocols	AODV, DSDV, DSR, OLSR, Click, Nix-Vector,	AODV, GPSR, BGP, DYMO, OSPF, PIM, RIP	×	AODV, BGP, IGRP, BRP, DYMO, DSR, OLSRv2, STAR, ZRP, ODMRP, and lots more	DSR, AODV, OLSR, ZRP, BGP, IGMP, PIM	-

simulators.

A simulation environment with realistic wireless networking and vehicular traffic patterns with which to evaluate VANET standards and protocols for suitability of usage in the envisaged solution is required. The chosen simulators have to meet the requirements as stipulated in the last two sections.

INET provides models for IEEE 802.11p, signal propagation effects, the physical environment as well as various ad hoc routing protocols.

OMNeT++ provides scalability and comprehensive result analysis, whilst integration of vehicle mobility simulation with INET is possible within OMNeT++ by using Veins_INET and SUMO.

SUMO provides the option of accurate vehicle traffic simulation as well as creation of custom maps on which to simulate vehicle traffic, meeting the specified requirements.

Based on these statements, as well as the discussion in Section 2.1.4 and information presented in Table 2.4 and Table 2.5, OMNeT++, INET, Veins_INET and SUMO are the simulation tools of choice since they meet the requirements set forth.

2.3 Hardware for VANETs

VANETs have not seen mainstream adoption as of yet, thus hardware availability for research purposes and commercial implementation can be expensive and lack variety. Apart from the automotive industry, companies such as uBlox, Redpine Signals, Arada Systems, and Codha Wireless have developed hardware that is dedicated to the VANET environment. They serve as excellent tools to aid in the development and research in the VANET field. These devices and modules can be costly. Many researchers that do not have the support from automotive companies or funding from corporate or institutional projects, have resorted to modifying existing hardware to suit their needs (as discussed later in this chapter). Firstly, the commercially available units will be discussed, and then the options of utilising existing hardware.

2.3.1 Commercially Available Hardware

Redpine Signals provide comprehensive solutions for V2V and V2I communications. This includes an OBU, RSU, WaveCombo chipset and module [61]. The OBU and RSU are ready-to-deploy units, whilst the WaveCombo chipset and module are intended for custom hardware development. Units support IEEE 802.11a/b/g/n/p, IEEE 1609, ETSI ITS-G5, Bluetooth 4.0 and have dual radios. A comprehensive SDK is also provided to aid development and testing.

Arada Systems provide a plethora of VANET and ITS hardware [62]. This includes the LocoMate OBU (various versions), LocoMate RSU (various versions), LocoMate ME and more. However, in this thesis only the LocoMate Classic OBU and LocoMate RSU-200 will be discussed. Both units provide IEEE 802.11p, IEEE 1609, dual radio and GPS support. An excellent Linux based SDK with C libraries is provided for development and testing.

Cohda Wireless also provide ready-to-deploy OBU and RSU units, called MK5 OBU and MK5 RSU respectively [63]. Both units support IEEE 802.11p, IEEE 1609 and ETSI ITS-G5. The units also sport NXP chips with custom Cohda firmware, dual radios, rugged enclosures and an accurate GPS implementation. A SDK called CohdaMobility is available for Linux operating systems, which can be used to customise and tune the module to user specifications. It also provides a virtual machine that allows the development and testing of applications without the need of the physical MK5 units.

uBlox is currently developing the THEO-P173 and VERA-P1 series modules that are designed for V2V and V2I communication [64][65]. uBlox does not provide a deployable OBU, RSU or evaluation kit at the time of writing, although an evaluation kit is planned for the VERA series. IEEE 802.11p,

IEEE 1609 and ETSI ITS-G5 standards are supported by the modules, each with a single radio. No SDK or development software has been announced yet.

Table 2.6 compares the features and specifications of the abovementioned hardware solutions. WaveCombo, LocoMate and MK5 hardware solutions provide the most lucrative feature set as well as software support. LocoMate and MK5 units are out of the box ready-to-deploy systems, whereas uBlox and WaveCombo solutions are modules which can form part of a custom system. In the case of this project - hardware level development is outside of the scope. This makes the Arada LocoMate and Cohda MK5 units the best choice.

Table 2.6: Comparison of commercially available hardware

Features	**Hardware Option**				
	WaveCombo Hardware	LocoMate Classic OBU and RSU-200	MK5 OBU and RSU	THEO-P173	VERA-P1
Supported Standards					
IEEE 802.11p	✓	✓	✓	✓	✓
IEEE 1609	✓	✓	✓	✓	✓
ETSI ITS	✓ (included in SDK)	×	✓	✓	✓
ARIB STD-T109	×	×	×	×	×
Additional Features					
Dual Radio	×	✓	✓	✓	✓ (P173 and P174)
GPS	✓	✓	✓	×	×
SDK	✓	✓	✓	×	×
Sample Applications	✓	✓	-	×	×
Ready to Deploy	✓	✓	✓	×	✓ (Planned evaluation kit)
Price	-	USD900 and USD1399	EUR1880 (OBU)	-	-

LocoMate Classic OBU does unfortunately not support ETSI ITS standards, but it is more than 50% less expensive than the MK5 OBU. However lucrative these options may seem, the steep price will make deployment of even a few units improbable in the case of this project.

If 'large scale deployment' is defined as 25 active nodes or more, the minimum cost of deployment would be USD 22 500 (about ZAR 320 000) when the LocoMate Classic OBU is chosen. Even in the case of merely 5 nodes the cost of deployment would be about ZAR 64 000. In the case of this project a maximum of ZAR 15 000 can be spent, ruling out large scale deployment as a whole, even with inexpensive hardware. It is not possible to obtain 25 units at ZAR 600 each, thus simulation is the only remaining option for large scale evaluation.

2.3.2 Custom Implementation of Existing Wireless Hardware

As seen in Table 2.6, commercially available VANET communication hardware can be expensive. However, some researchers have implemented VANET network protocol stacks on existing hardware in an attempt to reduce project cost or due to lack of commercially available hardware at the time.

González *et al.* [66] experimentally evaluated the suitability of the IEEE 802.11b standard for single-hop V2V communication. At that time, VANET-specific wireless standards such as IEEE 802.11p had not been drafted yet and not many studies using IEEE 802.11b for V2V communication were done. The authors used a Mikrotik RouterBoard 532A embedded computer paired with a Ubiquiti SuperRange2 802.11b/g mini-PCI module as their experimental hardware setup. It is mentioned that a “tiny Linux distribution" was used as the operating system of choice, although no specifics were provided. After experimentation it was concluded that IEEE 802.11b is suitable for V2V communication in urban and high-mobility scenarios, albeit performance was severely negatively affected in Non-Line of Sight (NLOS) conditions.

[67] performed multi-hop VANET communication experimentation in both urban and highway scenarios. Their research was motivated by the absence of publications on performed real-world multi-hop VANET experimentation due to hardware cost, routing protocol implementation and logistic difficulties. A Soekris Net4521 embedded computer in conjunction with a Texas Instruments ACX111 802.11b/g mini-PCI wireless card was used as OBUs in four vehicles. The operating system of choice was Linux Voyage with kernel 2.6.22, and OSLR routing protocol was implemented. With this experimental configuration as basis, the authors determined that achievable communication range between vehicles, and that OSLR routing protocol performance, is in practice significantly different to what was portrayed in previous literature.

An open-source, modular and low cost solution to evaluate IEEE 802.11p was proposed and evaluated by Agafonovs *et al.* [68]. The proposed system used a PC Engines ALIX2D2 embedded computer fitted with a Unex DCMA-86P2 802.11p mini-PCI wireless card to create an OBU and Road Side Unit (RSU). OpenWRT Linux distribution was used as the operating system, with modifications applied to the *at5k* driver in order to enable IEEE 802.11p operation for the wireless adapter. Large, high gain external omnidirectional antennae were used which resulted in reliable communication over large distances, given ideal antenna positioning and Line of Sight (LOS) conditions. Testing of single-hop LOS communication was performed in static and mobile conditions to evaluate the hardware setup and IEEE 802.11p implementation.

Barcelos *et al.* [69] created a system to report internal vehicle information via VANET to a remote server by making use of IEEE 802.11p for V2V communication. Multi-hop data transfer between vehicles in a small urban area was evaluated in LOS and NLOS conditions. Mikrotik RouterBoard RB411U embedded computers were used for OBUs whilst the RB443AH model was used for the RSU. Unspecified mini-PCI cards were used to enable wireless communication after driver modifications were performed in the operating system. OpenWRT was the operating system of choice, with an implementation of BATMAN routing protocol to facilitate multi-hop communication. The authors concluded that this hardware setup yielded satisfactory results, obtaining a Packet Delivery Ratio (PDR) of more than 80% for distances of up to 200 m.

Vivek *et al.* [70] created and implemented the IEEE 1609 WAVE stack on a Linux based Gateworks Ventana GW5400 embedded computer. A SparkLAN WPEA-128N mini-PCIe wireless module was used with modification to its drivers in the operating system, and various modifications were made to the wireless network subsystem to implement IEEE 1609 WAVE communication. Specific Linux distribution or kernel version is not mentioned. The authors successfully achieved interoperability of their system with commercially available hardware from Arada, and mentions that their solution provided superior performance in some aspects such as PDR during experimentation.

Various other researchers, such as Abunei *et al.* [71], also made use of Linux based embedded computers in conjunction with various mini-PCI or mini-PCIe wireless network cards to create an open-source, customisable hardware testing platform.

Low-power embedded computers that can use 12V DC as an input voltage is common throughout the referred research papers. This choice is logical given the fact that an OBU is intended to be powered by the vehicle itself, which provides a 12V DC power source.

All the discussed solutions made use of mini-PCI or mini-PCIe wireless adapter modules. In cases where driver modification was needed to force operation in the IEEE 802.11p spectrum, wireless modules with chipsets that have open-source drivers were used. It was seen that these wireless adapters contained Atheros chipsets that use *ath5k* or *ath9k* drivers which are open-source and natively supported by Linux. *ath5k* is pre-installed in Linux kernel version 2.6.25 and later [72], whilst kernel version 2.2.27-rc3 and up contains the *ath9k* drivers [73]. Another useful aspect of these wireless modules is the fact that external antennas of choice can be connected to suit the needs of a specific scenario.

It was also seen that variants of the Linux operating system was used in all solutions, and is assumed to be the case due to it being most popular operating system supported by embedded computers, with community support and development. It is also lightweight, open-source, highly customisable and

provides native driver support as mentioned earlier. Many routing protocol implementations are also built for the Linux environment.

It was clear that the two key components required to create a custom solution, with which to test IEEE 802.11p based VANET communication, is: 1) An appropriate wireless adapter and 2) a compute platform that supports said adapter. The wireless adapter will need to have the following features, based on previous literature:

- Capable of operating in the 5 GHz band
- Should use ath5k or ath9k open-source drivers
- mini-PCI or mini-PCIe form factor (with support for external antenna(s))
- Capable of ad hoc operation mode

Commercial wireless adapters containing Atheros chipsets that use ath5k and ath9k drivers are listed by [74] and [75] respectively. These sources are however outdated (updated in 2015), but it does provide a decent starting point. By looking up specifications of these listed devices, it was determined that the ath5k driver and devices using the driver does not support operation in the 5 GHz band, therefore eliminating that option. A list of Atheros chipsets using ath9k drivers that support operation in the 5 GHz band is presented below in Table 2.7, based on information provided by [73] and [75]:

Table 2.7: Atheros chipsets using ath9k drivers

AR9160	AR9220	AR9280	AR9380	AR9382
AR9340	AR9462	AR9580	AR9550	

As far as ad hoc mode is concerned - ath9k, and by extension the chipsets listed above, provide support for this mode of operation [76]. Thus it is concluded that any mini-PCI or mini-PCIe wireless adapter implementing one of the above mentioned chipsets will meet all the necessary requirements. Since there is such a wide range of devices to choose from, some of which can be seen in [75], specific selection will be determined by cost and availability.

In lieu of the reviewed documentation, it follows that the processing platform choice is limited to an embedded computer. Essentially, the compute platform only needs to meet two requirements: Have at least one mini-PCI or mini-PCIe expansion slot and should be capable of running a Linux distribution with kernel version 2.2.27 or higher. It is not required to support 12V DC input since converters or inverters can be used to meet the necessary power requirement

specification. Based on these requirements, devices such as laptops, compact all-in-one computers, and single-board embedded computers would all fit the scope depending on their specification.

The large choice of suitable options that satisfies the selection criteria rules out table a format comparison. Pricing, availability and the choice of wireless adapter (mini-PCI or mini-PCIe slot) will dictate the computer platform decision.

2.4 Other Related Work

Most of the research in the VANET field is focussed on developing and improving the wireless communication technology that enables VANET functionality. Research that propose and evaluate potential applications for VANETs mostly involve safety-related applications. This is understandable since it is probably the most enticing use-case for VANETs. As this project aims to assess the viability of a non-safety related application, there are not many similar works.

Llorca *et al.* [77] proposed a vehicle detection system to aid the collection of floating car data in the context of VANETs. Cameras mounted on public transport vehicles extract the number of vehicles located around the host vehicle, and their velocities. Data is transmitted to a central unit via a GPRS/UMTS connection. This approach does however not make use of inter-vehicle communication to aggregate the data.

Khaitiyakun and Sanguankotchakorn [78] proposed a technique based on Content Delivery Network (CDN) to disseminate data in a VANET. For routing, the OLSR protocol was used, whilst CDN candidates were selected from Multi-Point Relay in OLSR. They concluded that the CDN approach provides a substantial improvement in successful data delivery rates over the 'vanilla' OLSR implementation.

Piñol *et al.* [79] evaluated video streaming in VANETs with the Veins framework in OMNeT++. The simulation was performed in a urban environment - real world map data was imported from OpenStreetMap. They observed that the obstacles in the urban environment greatly affects packet delivery rates, with measured packet loss rates from 25% to 51% depending on vehicle routes.

Rezende *et al.* [80] designed a video-dissemination protocol called Reactive, Density-Aware and Timely Dissemination protocol (REACT-DIS) for implementation in VANETs. Network Coding was used to increase redundancy and hence decrease packet loss. REACT-DIS proved to deliver suitable end-to-end latency for video streaming purposes, whilst the implementation of Network

Coding lead to the fulfilment of delivery ration requirements.

M. Al-Tahrawi *et al.* [81] compared the performance of AODV and OLSR routing protocols in Hybrid sensor and vehicular networks. The simulation was performed in a rural environment, with the dissemination of safety related information. However, only six nodes and one RSU was used in the simulation. They concluded that AODV outperforms OLSR in this specific scenario.

Noori and Valkama [82] proposed a method to reduce the travel time of vehicles in an urban environment by using VANET technologies. The method aims to provide the current travelling time for each street as well as monitoring the real-time traffic conditions of each street. The simulations were performed with OMNeT++ and SUMO in a large-scale urban area.

2.5 Challenges

Various challenges and potential difficulties exist in the VANET research area. These challenges are present due to the unique characteristics of VANETs itself, and because it is a relatively new area of study. Thus, VANET specific technologies and standards, simulation tools and physical hardware availability each pose unique challenges.

Although many new wireless communication standards for VANETs have been developed and adopted in the past few years, there is yet no widespread adoption of such technologies in the public space. Thus, the question remains regarding the performance and possible shortcomings of these technologies when it comes to large-scale deployment. As more and more VANET applications are implemented around the world, light will be shed on currently unseen problematic areas, providing an opportunity to improve and optimise the technologies.

Currently, the focus of VANET technologies is on safety related applications, which usually only need to disseminate data to vehicles in close proximity. Depending on the network density, congestion and flooding can be a major issue that needs to be kept in mind when developing these applications, since delivery time of safety messages are of critical importance. When multi-hop data dissemination is needed, selection of the appropriate routing protocol is essential. It is known that frequent link breakages will occur, which can greatly impact network performance and delivery times if enormous amounts of control messages flood the network. This also holds true for non-safety applications. Although non-safety data may not be of time critical nature, a fine balance should be struck when it comes to prioritising data and channel access to ensure reliable delivery of safety messages, even when various non-

safety applications make use of the same devices and medium to transfer data. Strategic placement of RSUs and applications that leverage RSUs have the potential to combat some of these challenges.

As mentioned earlier, hardware availability and pricing is currently a major challenge for real-world viability testing of VANETS due to the technologies still being in a stage of infancy. Various researchers have turned towards creating custom solutions, but everyone has a unique solution and do not provide sufficient detail regarding the setup and configuration. Some of the components are outdated or niche, and can be hard to obtain. An affordable, open-source, available, and ready-to-use custom hardware solution implementing IEEE 802.11p and a routing protocol would be a great addition to the field of real-world VANET research.

Security within a VANET is a part that receives little attention. It is of critical importance, as in all other digital communication. Security flaws can be utilised by malicious attackers to alter, spoof or deny communication. As mentioned by the authors of [83], Sybil attacks, denial of service attacks, alteration attacks, and fabrication attacks can all greatly impact network performance, stability, reliability, and trustworthiness of data. Access to affordable hardware for field tests and scalability testing can also be a barrier in VANET research. As for simulation environments, there is no single, comprehensive tool available for dedicated VANET simulations. Most tools lack functionality or features in certain areas, which can hinder the development and testing of applications for VANETS.

Chapter 3

Simulation Environment Realisation

Ideally, commercially available VANET products and hardware would be installed in vehicles and used to perform 'real-world' evaluation of different networking standards and technologies to determine suitability for the desired application. It is assumed that a large number of installations, at least 25 to 50 vehicles, would be needed to obtain a clear picture of the realistic performance that can be expected from such a system.

However, due to the limited availability and cost of commercial hardware solutions, it would not be feasible to approach the solution in this manner. Even creating a more affordable custom hardware implementation to deploy on a medium to large scale would still not fit within the budget of this project, as portrayed in Chapter 2. Using simulation is the only remaining option.

Both network and traffic simulation would be needed to create a realistic scenario since network performance and characteristics would be heavily impacted by positioning and mobility of vehicular nodes. Network and traffic simulation would need to be run in parallel to create a dynamic simulation environment.

This project aims to build and evaluate a functioning solution with existing VANET standards and protocols in a simulation environment. The objective of this study is implementing and evaluating chosen protocols and standards for suitability in the desired application scenario and not comparing all available VANET technologies and protocols in detail. Strict adherence to standards is not an explicit requirement; therefore adjustment of parameters could be performed to investigate if deviation in parameter values would result in a more suitable or better-performing solution.

Setting up a fairly accurate and representative real-world scenario simulation environment would aid in providing insight into performance numbers and is-

sues that could arise in future deployment of such a solution. The main goal is the application of the chosen tools within allocated time constraints, and not the creation of the best possible simulation environment in order to most accurately model the real world.

The choice of network simulation tools would set the boundary for the protocols and standards that could be used in this project, since the objective is not to create a new protocol or simulation model, but rather to use existing models and implementations to create the desired solution. The choice of network simulation tool will restrict the VANET standards and protocols used. Traffic and network simulation have to be integrated as mentioned, which will influence the traffic simulator choice.

A Combination of OMNeT++, INET, SUMO and Veins_INET will be used to realise the simulation environment, as discussed in Chapter 2. Choosing appropriate simulation models to model effects such as signal fading and implementation of physical obstructions is an important part of creating a simulation scenario that provides the best possible representation of a real-world scenario within the bounds of the chosen network simulation tools, and is discussed in this Chapter.

3.1 Choice of Simulation Environment

Network simulation can be valuable in investigation, analysis and to hypothesise the behaviour of various possible network scenarios. Results obtained from a well-constructed simulation setup can prove to be crucial when designing a network for an unknown environment, or verifying theoretical results when creating a new standard or protocol. Simulation does, however, have its limitations - accuracy and trustworthiness of simulation models and scalability are only a few of the barriers to face along the way. Thus, network simulation software might not be the optimal tool to use for each instance of networking research, but it can be a great stepping stone.

Simulating a VANET environment requires simulation of vehicle mobility as well as the networking aspect. Chapter 2.3 provides the basic requirements that vehicle mobility and network simulators need to satisfy for successful simulation of a VANET environment. This chapter also provides an overview of various simulation platforms, and explains why the combination of OMNeT++, INET, Veins, SUMO and OpenStreetMap would be the optimal combination of simulation environments. Each of these software packages are widely recognised by their respective communities, and they seem to have great synergy, integration capabilities, as well as customisation possibilities. OMNeT++ will provide the main simulation framework and perform the network simulation,

whilst INET and Veins can both provide the required suite of models such as the PHY and MAC network layer models that are executed by OMNeT++. It should be noted that Veins and INET's model suites are independent of each other, resulting in a choice between them. Veins can be used without including INET, but if INET is the framework of choice, Veins must be used to act as the link between the mobility and network simulations, and is discussed next. Mobility simulation will be performed by SUMO, which has the ability to import road networks from OpenStreetMap map data. Veins and SUMO can be bi-directionally coupled via a TCP socket and communicate with the Traffic Control Interface (TraCI) protocol. Figure 3.1 provides a simplified overview of the components and integration of Veins with OMNeT++ and SUMO. The network and traffic simulations can run perfectly in parallel, and data obtained from SUMO can be used to update mobility characteristics of corresponding nodes in the network simulation as required. With both aforementioned packages VANET simulation setup can be approached either by using the model suite of Veins or INET.

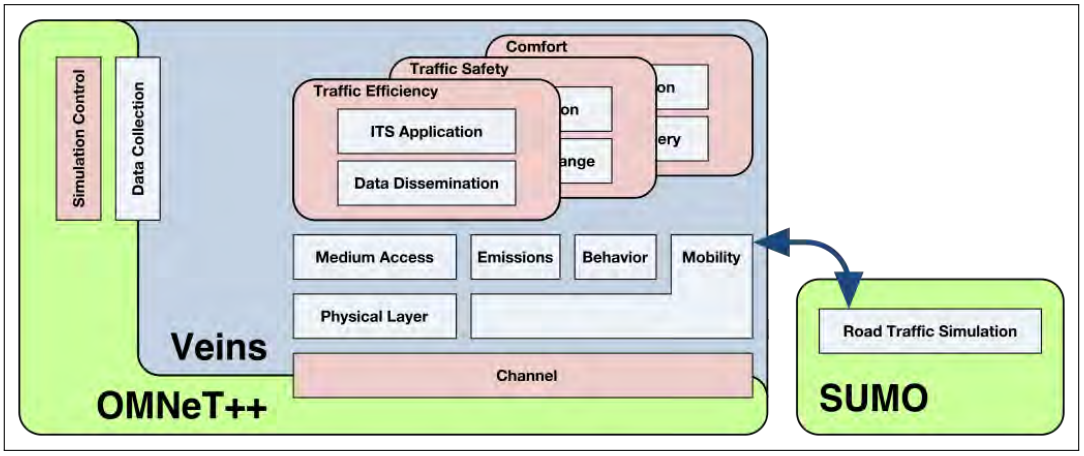


Figure 3.1: Simplified integration of OMNeT++, INET, Veins_INET and SUMO. Source: `https://veins.car2x.org/documentation/veins-arch.png`

As seen in Chapter 2.2, INET provides models for various routing protocols like AODV, whilst Veins does not include any routing protocol models. On the other hand, the Veins framework provides models for all the popular extended MAC standards used in the ITS space, e.g. IEEE 1609 and ETSI ITS-G5. Models for these standards are not included as part of the INET framework. Both frameworks provide models for aspects such as signal propagation effects, IEEE 802.11p, obstacle shadowing and antenna patterns. It should thus be asked if routing protocol models or the various ITS MAC models would be more beneficial for this project. The ideal scenario would have been the ability

to implement routing protocols in conjunction with one of the modified MAC models, but unfortunately it is not possible without significant effort and time - which can be a project on its own. Given the scenario and known goals for this project, framework choice would be aided by the following:

- Creating and implementing routing protocol support in Veins from scratch is not one of the project's goals. The idea is to use / modify existing tools, frameworks and standards.
- Hypothetically, it would be possible to create the envisaged system without explicitly implementing IEEE 1609, ETSI ITS-G5 or AIRB STD-T109 and merely use the IEEE 802.11p standard as-is.
- The envisaged application does not fall into one of the standard VANET application categories, and will not form part of or contribute to VANET specific goals like real-time traffic management or reporting of safety-related incidents.
- Processing of the data will not be performed by the nodes themselves, the data merely needs to get to the nearest aggregator. Intermediate nodes only need to act as hops or temporary storage.
- If the goal was to implement the envisaged system as part of an existing ITS network, it would have been crucial to adhere to the standards being used in such a network. However, this is not part of the project's goal and any standard(s) of choice may be used.
- Implementing an appropriate routing protocol in this scenario could potentially increase the network coverage area and enable multi-hop data dissemination. This would greatly aid data aggregation.
- As mentioned in Chapter 2, South Africa does not yet provide guidelines as to which standards / spectra are to be used for ITS applications.

Ideally one also needs to compare models that both frameworks provide, e.g. IEEE 802.11p, to choose the most suitable option. The implementation of similar models in Veins and INET are of course not identical, and it is very difficult to accurately compare them by merely reading the documentation, official publications and by inspecting the code repositories. The creators of Veins published a paper for almost each model they created for Veins, which serves the purpose of providing motivation for and verifying the implementation of said models. This is of benefit, as it provides some credibility for these models. INET does not have the same amount of validation publications by the authors and due to constant updates and changes to the framework, it is rare to find an up-to-date publication. It seems as if the only way to accurately compare models would be to simulate identical scenarios and analyse

the obtained results. However, to what trusted benchmark would these results be compared to know which framework provides a more accurate depiction of the expected real-world performance?

Considering the the points made above, and knowing that data aggregation is one of the most important aspects for this project, and it can be intuitively reasoned that having the option to implement and test routing protocols would be more beneficial than to explicitly implement either IEEE 1609, ETSI ITS-G5 or AIRB STD-T109. Since they all extend IEEE 802.11p, it is assumed that only IEEE 802.11p is needed to implement the solution. The solution is also envisaged to be a standalone application and is not intended to form part of an ITS ecosystem, further motivating the point that an extended standard is not necessary to implement the solution.

3.2 Initial Setup of OMNeT++, INET and Veins

The following section provides a description of how the simulation environment was initially set up. During this time, Veins 4.6 was the latest release of the Veins framework, and provides the 'Veins_INET' project. Package version constraints for this release of Veins dictated which versions of OMNeT++, INET and SUMO had to be used. Details of the software versions used at this point, as well as the hardware used for the simulation testbed, are provided in Table 3.1.

Since OMNeT++ was developed to run in a UNIX environment, MinGW was installed to accommodate OMNeT++ in a Windows environment. OMNeT++ was now built by pointing to the source files in the MinGW shell, and running `./configure` followed by `make -j12`. After the make has completed, the installation was verified by calling omnetpp to open the OMNeT++ IDE, and running a few of the included sample projects.

Next, the INET framework was imported and built. The installation was verified by running the simulations contained in the 'tutorials' folder, and making sure the outputs were identical to those found on the official INET Wireless Tutorial website. After importing and building Veins framework's source in OMNeT++, the included demo scenario can be used as a tool to verify the TraCI communication between the network and mobility simulations. It was confirmed that SUMO is working correctly by executing the provided SUMO demo scenario, erlangen.sumo.cfg, with sumo-gui.exe.

Enabling TraCI communication between SUMO and the networking simula-

Table 3.1: Network simulation testbed specifications

Testbed Hardware	
CPU	Core i7-8700k (6 cores, 12 threads)
RAM	16GB DDR4 3000Mhz
GPU	Nvidia GTX 1080
Storage	8TB HDD, 250GB SSD
Operating System	Windows 10
Simulation Software	
Package	**Version**
OMNeT++	5.3.2 (stable release)
INET	3.6.0 (stable release)
Veins	4.6
SUMO	0.30.0

tion in OMNeT++ is achieved by running a daemon called sumo-launchd. This is achieved by executing the included sumo-launchd.py script prior to starting the networking simulation. This daemon listens for incoming TCP connections from the TraCIScenarioManager module in Veins, and creates instances of SUMO simulations on request. Parameters required to setup the SUMO simulations are sent by TraCIScenarioManager to the daemon as the networking simulation is started in OMNeT++. Requests are proxied between SUMO and OMNeT++ for the duration of the networking simulation, after which the daemon destroys the SUMO instance. An overview of the integration and communication between the various components comprising the simulation environment is shown in Figure 3.2.

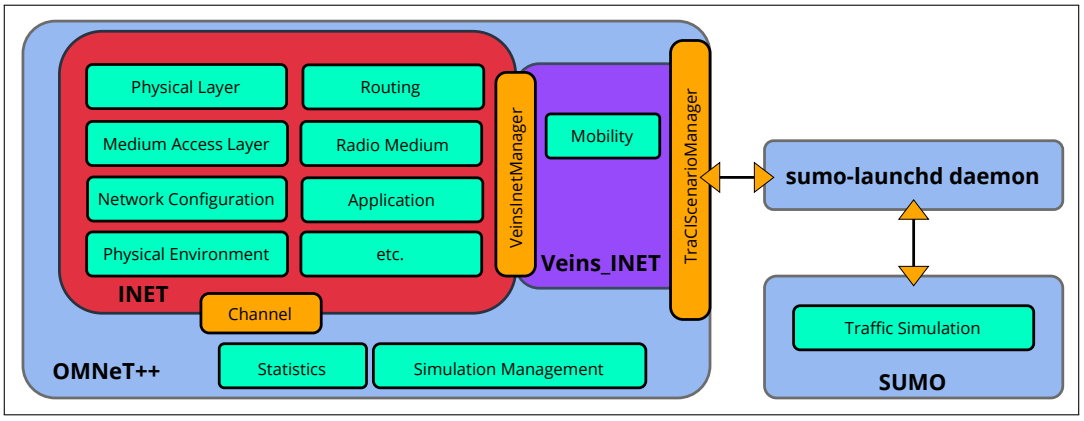


Figure 3.2: Integration and communication between OMNeT++, INET, Veins_INET and SUMO. Inspiration drawn from Figure 3.1

After starting the daemon and running the Veins demo simulation, it was

confirmed that the output of the simulation matched the output seen on the official Veins tutorial website. To further confirm that positional information is correctly communicated to OMNeT++, a visual inspection was performed by comparing the position of vehicles in both simulation environments at identical, discrete timestamps.

Lastly, the functionality and integration between INET and Veins_INET was required. At its core, Veins_INET allows Veins to be used purely as a mobility model and node manager in the INET framework. Vehicle nodes are modelled by inheriting from the INET WirelessHost module, which models a host with the full networking stack. This model features a configurable 802.11 wireless card containing the MAC and PHY layers, IPv4 for the network layer, transport layer with TCP, UDP and SCTP, as well as application layer support for each transport layer protocol.

VeinsInetMobility (a module from the Veins framework used to describe the positioning information of vehicular nodes) is then used as the mobility module for the nodes, and INET's HostAutoConfigurator module is used for automatic network configuration. A global module called VeinsInetManager is responsible for the TraCI communication and management of vehicles in the networking simulation (see Figure 3.2)

VeinsInetManager interacts with the VeinsInetMobility module to update the mobility characteristics of each node during the simulation. Veins_INET provides a small demo simulation, which was used to verify that the positional information was correctly communicated between SUMO and the networking simulation by using the same methodology as with the Veins-only simulation.

3.2.1 Set Up of IEEE 802.11 PHY and MAC

The existing components and models available in the previously chosen simulation frameworks / software will be used to create the simulation environment for this project. It should be made clear that no low-level modules or models were coded from scratch, but components were assembled from the various building blocks made available in the chosen frameworks.

Setting up the simulation can be divided into four main parts. 1) Composing models for the desired node types in the simulation and configuring the lower layers of the networking stack, whilst keeping the quantity of nodes low and their mobility static. 2) Configuring the upper layers of the networking stack, i.e. routing protocol and application layer. 3) Introduction of mobility, obstacles (i.e. buildings) and increasing the number of nodes to determine possible optimisations for larger scale simulations. 4) Setup of appropriate

result recording and analysis tools.

Basic models need to be composed in order to create the wireless nodes needed in the simulation. Three types of nodes are required: Vehicles equipped with a networking stack and a vehicular mobility model, vehicles without networking capabilities but with a vehicular mobility model and RSUs with the same networking stack as the equipped vehicles but without a mobility model (i.e. they will be stationary). The example node model provided by the Veins_INET project is used as a starting point.

The most basic form of these models would be a node that simply extends the WirelessHost module. As mentioned previously, WirelessHost is a compound module comprised of various individual sub-modules. By default, WirelessHost implements a configurable IEEE 802.11 wireless NIC set to STA mode. Ieee80211Nic module contains the following layers: PHY (modelling of the radio), MAC and management layer.

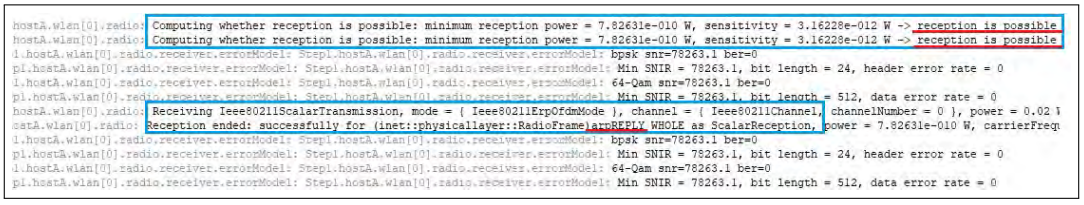
The radio module used by this NIC models the physical characteristics of the radio channel, and determines if transmissions were successfully received. Transmissions are modelled as analogue signals with scalar transmission power - hence the symbol, bit and sample domains of transmissions are not represented by this model. Successful reception of a transmission is determined by the ratio between the received signal's power compared to the sum of the power of other signals present in the medium. This ratio is known as signal-to-noise-plus-interference ratio (SNIR or SINR). If SNIR minimum is sufficiently high for the duration of the transmission, reception is considered to be successful. Calculation of bit error rate (BER) based on the SNIR and radio operation mode (BPSK, QPSK, 16QAM, etc.) is then performed to determine if frames suffered bit errors during the transmission. By default, Ieee80211NistErrorModel module is used for the BER calculation. Frames are passed up to the MAC layer after confirmation of transmission success.

The Ieee80211Mac module is implemented to model the MAC layer. As mentioned in the INET documentation, this module is based on the 2012 revision of the IEEE 802.11 standard. For convenience of experimentation and testing, major components and policies of the MAC module, such as coordination functions, contention mechanisms, rate control algorithms, etc., are separated into individual sub-modules. These modules can be customised and replaced as required, thus making it easy to modify the MAC layer to suit the desired implementation.

The management layer exchanges the probe or beacon management frames. This layer also performs encapsulation of MAC data packets, channel scanning, association and handovers. Ieee80211MgmtSta is the default management

module implemented by WirelessHost (since it is set to STA mode by default), but can be replaced by Ieee80211MgmtAp or Ieee80211MgmtAdhoc depending on the requirements. Ieee80211MgmtAdhoc is intended to be used for ad hoc implementations. The main difference between this module and its STA counterpart is that it does not handle authentication, association, beacon frames or probe frames and discards any such frames received. Ieee80211MgmtAp is intended to be used when modelling access points, as the name suggests. It can be used to model an AP with wireless and Ethernet interfaces, but unlike Ieee80211MgmtSta, it does not perform channel switching.

A simple simulation containing two WirelessHost nodes equipped with Ieee80211-MgmtAdhoc management module and an UDP application layer is used as the starting point. One node acts as a source, sending UDP data packets to the second node which acts as a sink. In the network configuration file, IPv4NetworkConfigurator is used for IP address configuration and static route table initialisation at start-up, and Ieee80211ScalarRadioMedium is set as the radio medium module (since INET's other 802.11 models require the use of a 802.11 radio medium model). Figure 3.3 demonstrates the successful communication between the nodes. Firstly it is determined if reception is possible, then address resolution is performed by the Arp module, after which the data packets are sent. It can be seen that an ACK is sent after successful reception of a frame.



Src/Dest	Name	Info
hostA --> hostB	arpREQ	ARP req: 145.236.0.2=? (s=145.236.0.
hostB --> hostA	arpREPLY	ARP reply: 145.236.0.2=0A-AA-00-00-(
hostA --> hostB	ACK	WLAN ack 0A-AA-00-00-00-02
hostA --> hostB	UDPData-0	inet::ApplicationPacket:1000 bytes
hostB --> hostA	ACK	WLAN ack 0A-AA-00-00-00-01
hostA --> hostB	UDPData-1	inet::ApplicationPacket:1000 bytes
hostB --> hostA	ACK	WLAN ack 0A-AA-00-00-00-01
hostA --> hostB	UDPData-2	inet::ApplicationPacket:1000 bytes
hostB --> hostA	ACK	WLAN ack 0A-AA-00-00-00-01
hostA --> hostB	UDPData-3	inet::ApplicationPacket:1000 bytes
hostB --> hostA	ACK	WLAN ack 0A-AA-00-00-00-01

Figure 3.3: Successful reception and communication

After verifying that the simplest form of the simulation works as expected, the radio model's parameters is changed to comply with the IEEE 802.11p

Figure 3.4: Extract from Ieee80211OFDMMode.cc

```
const Ieee80211OFDMMode Ieee80211OFDMCompliantModes::
ofdmMode6MbpsCS10MHz("ofdmMode6MbpsCS10MHz",
&Ieee80211OFDMCompliantModes::ofdmPreambleModeCS10MHz,
&Ieee80211OFDMCompliantModes::ofdmHeaderMode3MbpsRate5,
&Ieee80211OFDMCompliantModes::ofdmDataMode6MbpsCS10MHz,
MHz(10), MHz(20));
```

PHY specification. Parameters of interest are presented in Table 3.2, with the default and modified values. As mentioned in Chapter 2, the desired band of operation is 5.9 GHz, and the network traffic that will be generated will be non-safety related.

The choice of carrier frequency is dictated by channel specifications stipulated by IEEE 1609 WAVE (Figure 2.3) and ETSI ITS-G5 (Figure 2.3) for non-safety related applications. Both standards allocate the 5.855 - 5.875 GHz frequency range to non-safety related applications, which would be channel 172 and 174 in the case of IEEE 1609 WAVE and SCH3 and SCH4 in the case of ETSI ITS-G5. The carrier frequency was chosen to be 5.86 GHz which complies with WAVE channel 172 and G5-SCH4. A data rate of 6 Mbps was chosen since support for transmitting and receiving at this rate is mandatory for any IEEE 802.11p PHY implementation, and according to Jiang *et al.* [84] it is the optimal data rate for VANET applications. The creators of the Veins framework also used this data rate in some of their publications on VANETs, such as [4].

Confirmation that the model supports the chosen parameters was verified by inspection of the source files of the radio model (specifically in the Ieee80211-OFDMMode.cc file). Figure 3.4 is a one-line source-code extract from this file that specifies a pre-defined mode of operation. The extract indicates that an OFDM mode with 6 Mbps data rate and 10 MHz bandwidth is a mode of operation supported by the model.

Table 3.2: Modified radio parameters

Parameter	Default	Modified
**.wlan[*].radio.bandName	"2.4 GHz"	"5.9 GHz"
**.wlan[*].radio.carrierFrequency	2.412GHz	5.86GHz
**.wlan[*].radio.bandwidth	2MHz	10MHz
**.wlan[*].radio.channelNumber	0	1
**.wlan[*].bitrate	54Mbps	6Mbps
**.opMode	"g"	"p"
*.radioMedium.mediumLimitCache.carrierFrequency	2.4GHz	5.86GHz

Table 3.3: IEEE 802.11p data rates

Data Rate (Mbps)	Modulation	Coding Rate	Data bits per OFDM symbol	Mandatory to Support
3	BPSK	1/2	24	Yes
4.5	BPSK	3/4	36	
6	QPSK	1/2	48	Yes
9	QPSK	3/4	72	
12	16-QAM	1/2	92	Yes
18	16-QAM	3/4	144	
24	64-QAM	2/3	192	
27	64-QAM	3/4	216	

Investigation of the adjusted parameters' effect on this basic simulation yielded expected results. Before the changes, mediumLimitCache calculated the maximum communication range and maximum interference range to be 790.52 m and 14057.70 m respectively, which was reduced to 323.76 m and 5757.42 m respectively after the parameter adjustments. It should be stated that no path loss or signal fading algorithms were implemented at this point, and that the transmission power was kept at the default value of 20 mW with a unity-gain isotropic antenna model.

Data rate verification was performed by sending UDP data packets at an effective rate of 10 Mbps, which would saturate the channel if a data rate of 6 Mbps is in effect. Packets with 1000 B payload was sent every 0.8 ms for a total of 10 seconds. In the unmodified scenario, the destination node received all packets successfully, whilst only 5744 of the 12500 packets were received after parameter adjustments. This is an effective data rate of 4.595 Mbps, which correlates with findings presented in [24].

3.2.2 Implementing Signal Fading

Up to this point it has been assumed that all signals in the simulation propagate through free space with no obstructions or ground plane model. In a realistic environment, radio signals are affected by various elements such as reflection, absorption, and refraction. These effects causing signals' power to fluctuate, and decrease in density relative to the distance, is called signal fading. Path loss and shadowing both form part of signal fading. Path loss would be defined simply as the power decrease relative to the distance between sender and receiver, usually in a LOS scenario, whilst shadowing is defined as the fluctuations in signal power due to obstructions between the sender and receiver. Taking signal fading into account would be crucial in the pursuit to create an accurate representation of wireless communication in an urban area, especially since nodes could be sparsely spread out with large distances and various obstructions between them.

Table 3.4: INET path loss models

Path Loss Model	Description
FreeSpacePathLoss	Unobstructed line of sight path loss in a vacuum, derived from the Friis equation
BreakpointPathLoss	Dual slope path loss model with separate alpha parameters
LogNormalShadowing	An extension of FreeSpacePathLoss with an additional sigma parameter to model shadowing
TwoRayGroundReflection	Models interference caused by single ray ground reflection due to variation in the height of the sender and receiver's antennas in LOS scenarios. To implement this model, physicalEnvironmentModule with a ground plane needs to be added to the simulation
TwoRayInterference (INET 4.0)	Modification of TwoRayGroundReflection aimed at inter-vehicle communication
RayleighFading	Statistical model to model signal cancellation effects due to moving antennas and multipath propagation caused by obstacles. This model is most suitable for scenarios in which there is no dominant signal component (i.e. no direct LOS)
RicianFading	Similar to RayleighFading, but assumes a dominant signal component.
NakagamiFading	Refined model of RayleighFading and RicianFading. In cases where the shapeFactor parameter = 1, it is similar to RayleighFading, and when shapeFactor >1 results are comparable to RicianFading. This model tends to match empirical results better than other models in various scenarios.

For the purpose of staying coherent with terminology used by INET, all models used to calculate signal fading and power loss over distance will be called 'path loss models'. INET provides various path loss models, each with their own set of adjustable parameters. Table 3.4 provides an overview of the available models and their descriptions, with some information extracted from INET documentation [85]. Model names presented in Table 3.4, e.g. *FreeSpacePathLoss*, will be used throughout the rest of this section when referencing the specific INET models.

An appropriate path loss model needs to be selected, for the reasons stipulated earlier. However, selecting the most applicable path loss model at this point is not crucial, since modelling of an urban environment has not yet been introduced to the simulation. It would still be wise to investigate and compare the effects of the available models in an 'ideal' LOS scenario, because there could be rare instances where nodes will have LOS in an open space, even in urban areas. Introducing extra computation overhead caused by these models, is also a factor that needs to be kept in mind and investigated.

Path loss models, in this case, can be compared by network throughput and SNIR over distance. Thus, recording data in vector format would need to be set up for these two metrics. One method of recording time series data in OMNeT++ is making use of the simulation signal mechanism and the *statistic* property (denoted as @*statistic* in OMNeT++) in the applicable *NED* file. NED files are used to define and assemble simulation components (parameters, host models, statistic models, etc.) when building a simulation in OMNeT++. Signals and statistics can be declared in such a file to enable the capture and processing of data in the simulation at run-time.

Signals can be seen as 'event notifiers' which propagate up the module hi-

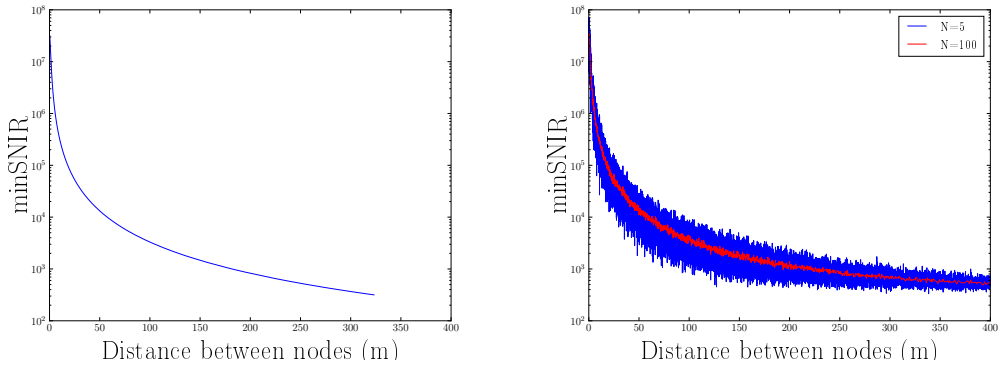
erarchy when emitted, each with their own name and optional value. Objects with callback methods, called listeners (in this case a statistic property declared in the NED file), can subscribe to a signal name and be used to process data received from that signal. The statistic property can be used to subscribe to a specific signal and use various predefined recorders (i.e. time series data or histogram recording) and filters (i.e. summing, computing the mean, computing the time average) to record and process data contained in the signal.

By using this method, time series logging of effective data throughput (thus the 'goodput') as well as logging of the minimum Signal-to-Interference-Plus-Noise Ratio (SNIR) value each time a packet is received by the radio is implemented. However, the desired metric is still value over distance, not time. By enforcing a direct correlation of distance between nodes and simulation time, the same time series data can be used to visualise values over distance. The simplest way to implement this would be to let the two nodes in the current simulation move apart in one axis of travel, at a constant speed of exactly 1 meter per second. Each second, or fraction thereof, would directly correlate to the distance between the nodes. This mobility characteristic in conjunction with the various path loss models was added to the simulation, with all parameters of the path loss models being left at default. Simulation time is extended so that the distance between the nodes would reach up to 400 m, since the theoretical maximum communication range was calculated to be 323.76 m as mentioned earlier.

Figure 3.6 provides a comparison between the goodput over distance for the available path loss models except for BreakpointPathLoss. No default values or documentation is provided for this specific model, thus it is not taken into consideration. It should be noted that a moving average with $N = 5$ samples was applied to the raw data due to large variations. Figure 3.7 showcases the raw values in comparison to the averaged values of the throughput results for NakagamiFading and FreeSpacePathloss.

FreeSpacePathLoss is the default path loss model implemented by Ieee80211-ScalarRadioMedium (a compound INET module used in conjunction with the IEEE 802.11 PHY layer model to model the medium through which signals travel) with systemLoss (parameter to account for hardware imperfections) set to 0 dB. As seen in Figure 3.6 and Figure 3.7, goodput remains constant with little variation over distance in the case of FreeSpacePathLoss. The same holds true for TwoRayGroundReflection, even when the height of nodes' antennas are varied. Another characteristic of these two models is the fact that no attempt at communication is made beyond the theoretical maximum communication distance as calculated by INET. This effect can be observed in Figure 3.6 as the turquoise 'FreeSpace' and green 'TwoRay' graphs abruptly cut off. Based on these results it can be argued that these two models do not

Figure 3.5: minSNIR over distance



(a) FreeSpacePathLoss Model (b) NakagamiFading model

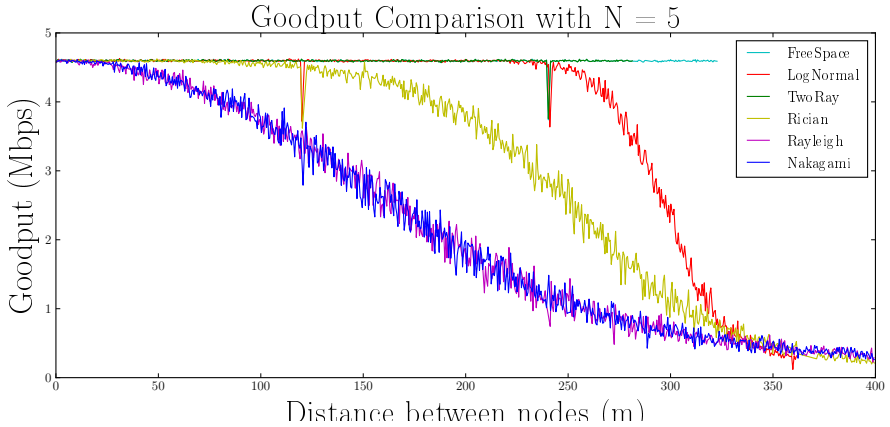


Figure 3.6: Goodput over distance for all models

provide an accurate representation of 'real world' performance.

In contrast, NakagamiFading, RayleighFading and RicianFading look like more suitable candidates. More aggressive declines in goodput over distance and less stable data transfer rates overall is more in line of what would be expected from empirical data, as seen in Figure 3.7 for NakagamiFading. It can be seen that NakagamiFading and RayleighFading provide the 'worst case' in terms of data transfer rates compared to the other models, with LogNormalShadowing being the most lenient in close range scenarios. NakagamiFading and RayleighFading yielded almost identical results - this is due to the fact that the default value of the *shapeFactor* property of NakagamiFading is 1, and confirms the information specified in Table 3.4. Given the observed results, choosing NakagamiFading as the path loss model to use would be a good option for now.

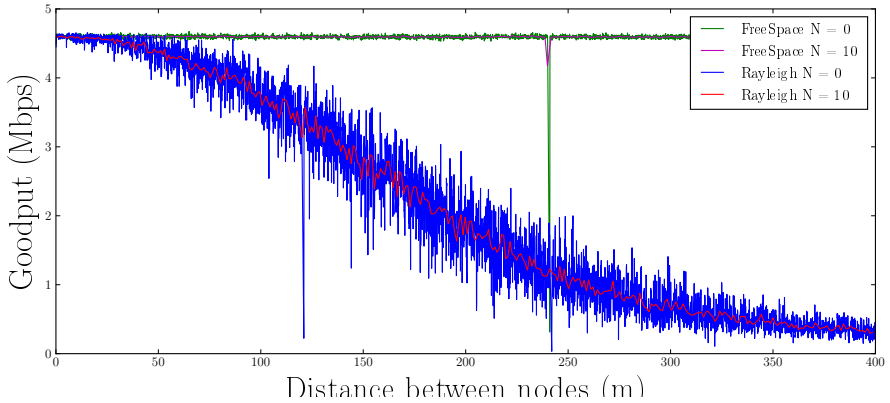


Figure 3.7: Raw goodput data vs window average

3.2.3 Implementing a Routing Protocol

Successful communication in a VANET is heavily reliant on the routing mechanism of the network, especially when multi-hop data transfer is required, as explained in Section 2.1.3. In this section it was also mentioned that VANETs have unique characteristics and challenges compared to other ad hoc networks. This can make it difficult to find an ideal routing protocol that will suit the needs of the application, since most protocols were not designed to account for such high node mobility and volatile network conditions.

3.2.3.1 Choosing a Routing Protocol

In Section 2.1.4 it was concluded that topology-based routing is the most suitable routing protocol type out of the types discussed in Chapter 2. It is, however, outside the scope to do an in depth analysis and comparison of all possible topology-based routing protocol models that might fit the use case. Designing a custom routing protocol or creating a new model from scratch is outside the scope of this project.

Therefore, the choice of a topology-based routing protocol will be made based on the availability of models in INET, as well as libraries or package availability for real-world implementation. Popularity of protocols in literature is also a factor to take into consideration, since it would be favourable to implement a well-studied and well-documented protocol in order to avoid possible unforeseeable road blocks, and to fall back on literature for reference if such an issue arises.

INET separates its available routing protocol models into two categories, namely 'Internet Routing' and 'Ad-hoc Routing'. Provided Internet Routing models are BGP, RIP and OSPF, whilst Ad-hoc Routing models are shown in Table 3.5. Furthermore, the table depicts popularity of each protocol in IEEE

Table 3.5: INET ad hoc routing protocol model comparison

Protocol	Category	Model Validity	Implementation easily available	IEEE Explore and ACM Digital Library search hits
AODV	Topology-based, Reactive	Based on RFC 3561	Yes	3202 and 221
DSDV	Topology-based, Proactive	Partial implementation	Yes	486 and 30
DYMO	Topology-based, Hybrid	Based on draft-ietf-manet-dymo-24	Yes	126 and 17
GPSR	Position-based	Not Specified	Not Specified	305 and 35

Explore and ACM Digital libraries. At the time of writing, a simple search was performed in both databases with the respective routing protocols' names as keyword, and the quantity of search results returned was taken as an indication of popularity. As seen in Table 3.5, one model for each topology-based routing protocol category is provided whilst one position-based protocol, namely GPSR, is provided. As mentioned earlier, GPSR can be ruled out since it does not fall within the category of interest. DSDV is also ruled out due to the model only being partially implemented. Being the two remaining options, AODV and DYMO both seem like suitable choices given the fact that both models are based on reputable IETF documentation, and both have usable real-world implementations. As depicted in Table 3.5, AODV has many more literature appearances than DYMO.

Angeles *et al.* [86] compared the performance of various ad hoc routing protocols under different signal propagation models in OMNeT++, with similar simulation parameters as this project. The authors' performance metrics of choice were packet delivery ratio and end-to-end delay over an increasing number of nodes in the simulation. Both AODV and DYMO formed part of the investigation, with NakagamiFading being one of the path loss models used. AODV and DYMO delivered similar results, with DYMO performing slightly better than AODV in all cases.

AODV and DYMO are both viable choices based on the information in Table 3.5 and [86]. DYMO did outperform AODV by a small margin in [86], but AODV does have a larger amount of supporting literature and available open-source implementation options. Due to the fact that the chosen routing protocol would need to be implemented in a real-world scenario in the later stages of this project, and given the larger amount of available literature, AODV is the routing protocol of choice in this case.

3.2.3.2 AODV Overview

Ad hoc On-Demand Distance Vector (AODV) was unveiled in 2003 by Perkins *et al.* [87] in the form of RFC 3651. It does not define a standard, but rather specification for implementation. AODV was designed for use in a large MANET, with key design aspects being rapid adaptation to link state changes and low network and processing overhead. Information presented in

this section is an interpretation of RFC 3651 [87].

Table 3.6: AODV route table fields

AODV Route Table Fields
Destination IP Address
Destination Sequence No.
Destination Sequence No. Validity Flag
State and Routing Flags
Hop Count
Next Hop
List of Precursors
Route Lifetime

Table 3.7: AODV message fields

RREQ Message Fields			
Type	Flags	Reserved	Hop Count
RREQ ID			
Destination IP Addr			
Destination Seq Num			
Source IP Addr			
Source Seq Num			

RREP Message Fields				
Type	Flags	Reserved	Prefix Size	Hop
Destination IP Addr				
Destination Seq Num				
Source IP Addr				
Lifetime				

RERR Message Fields			
Type	N	Reserve	Dest Cnt
Unreachable Dest IP Addr			
Unreachable Dest Seq Num			
Additional Unreachable Dest IP Addrs			
Additional Unreachable Dest Seq Nums			

Being a reactive routing protocol, routes are only created and maintained in the route table when necessary. A route table entry will contain information about the route to a specific destination, with applicable fields indicated in Table 3.6. AODV uses User Datagram Protocol (UDP) as the transport layer protocol, which contributes to low overhead and delay times, but at the cost of flow control and error recovery.

Route Discovery. When a source node desires to send data to a destination node, a route table lookup is performed to determine if an existing route to the destination is present. If a route exists, data is forwarded to the next hop as specified in the route table. Otherwise the route discovery process is initiated by broadcasting a Route Request (RREQ) message containing information indicated in Table 3.7. Before generating a RREQ message, the source node will increment RREQ ID by 1, since it acts as an unique identifier for the RREQ in combination with the source Internet Protocol (IP) address. The source sequence number is also incremented by 1 before initiation of the route discovery process. Sequence numbers play an integral role in maintaining up-to-date route information, which will be discussed throughout this section.

Upon receiving a RREQ message, a node will update its route table with a reverse route to the source node. However, the message will be discarded

in the case where unique identification fields of the RREQ match those of a previously received RREQ.

In the case of an intermediate node, it will check if it possesses a valid route to the destination when an unique RREQ is received. If so, a Route Reply (RREP) message with information specified in Table 3.7 will be unicast back to the source node, otherwise the RREQ will be re-broadcast after incrementing the *Hop Count* field by 1. A route entry is deemed to be valid if the destination sequence number (see Table 3.6) is equal to or greater than the destination sequence number contained in the RREQ.

In the case where a node receiving a RREQ is the destination node, it will update its sequence number and unicast a RREP back to the source node. The sequence number is updated by incrementing its current sequence number, or by replacing it with the destination sequence number contained in the RREQ message if it is greater than its own sequence number.

As a RREP propagates back to the source node, route tables of intermediate nodes will be updated with information contained in the RREP. Once the RREP reaches the source node, its route table will be updated and data transfer can be initiated.

Assessing Local Connectivity. Nodes may periodically assess the link status of active routes by making use of *Hello messages*. Hello messages, which are essentially RREPs with $TTL = 1$ (Time to Live), may periodically broadcast each *HELLO_INTERVAL* if a node forms part of an active route. Once a neighbour receives a Hello message, it should update the route lifetime and destination sequence number if the route exists, otherwise a new route table entry must be created. If a Hello message was received by a node within the past *DELETE_PERIOD*, and no subsequent Hello messages or other packets are received within *ALLOWED_HELLO_LOSS * HELLO_INTERVAL*, it is determined that the link is broken.

Connectivity is also determined by link layer notifications each time a packet is transmitted to the next hop in an active route. A link will be seen as broken in the following scenarios: Absence of CTS after transmission of RTS, no ACK received after a packet has been sent and lastly if the retransmission attempt limit is reached. If no link layer notifications are available, a node has to determine connectivity by monitoring the channel when the next hop is expected to forward a packet. If a transmission attempt is not seen within *NEXT_HOP_WAIT*, a RREQ or Internet Control Message Protocol (ICMP) Echo Request should be unicast to the next hop. The link will be classified as broken if no response is received.

Handling Link Breakages. After detecting a link breakage, a node will initiate Route Error (RERR) message processing. The node will invalidate the route and identify all destinations and neighbouring nodes affected by the link breakage. Destination sequence numbers of all affected destinations will

be incremented, after which a RERR message will be generated and sent out to all affected neighbours. This could be in the form of a series of unicast messages or a broadcast, depending on the amount of affected neighbours. All nodes receiving a RERR will invalidate the affected route in their route table and update the destination sequence number to that contained in the RERR message. The RERR should then be propagated via predecessor nodes until it reaches the source node, with each intermediate updating their route tables as mentioned above. Upon receiving a RERR, the source node may restart the route discovery process if necessary.

On the other hand, a node upstream of a link breakage may initiate a local route repair process. The node must increment the destination sequence number and broadcast a RREQ message for the desired destination. Whilst waiting for RREP messages all data packets should be buffered by the node attempting the route repair. If no RREPs are received, the RERR process discussed earlier will be initiated. In the case where one or more RREPs are received, and the hop count contained in the RREP is greater than the current hop count for the invalid route, a RERR with the 'N' bit set (see Table 3.7) should be generated and sent to affected neighbours. All nodes receiving this type of RERR should only update their route table with the information specified in the RERR message and not delete the route. This message must then be forwarded to other affected precursor nodes. This process may result in routes with larger hop counts, thus if such a message arrives at the source node it may decide to restart the route discovery process to find a shorter route.

3.2.3.3 Setup and Verification of AODV in INET

The ambiguous way in which RFC 3651 is presented may result in different implementations, depending on the reader's interpretation. Even though INET's implementation of AODV would be considered trusted, it would still be good practice to verify this by an experiment.

AODV routing is provided in INET in the form of the AODVRouting module. All default parameters match those recommended by RFC 3651 Section 10, with Hello messages and gratuitous RREPs disabled by default. Local route repair is, however, not implemented yet at the time of writing. Enabling nodes to use AODV in INET is made simple by setting each node to be an instance of AODVRouter, a WirelessHost module extending AODVRouting. The only requirement is to disable static route initialisation in IPv4NetworkConfigurator, since route tables will be managed by AODV.

To verify the functionality of AODV experimentally, at least three nodes are needed. The simulation used in Section 3.2.2 was used as-is, with a third stationary node being added 200m away from the stationary source node (which

is still in communication range according to Section Section 3.2.2). Nodes were changed to be instances of AODVRouter and Hello messages was enabled. As the mobile destination node moves away from the stationary source node, it will eventually be out of communication range and the intermediate node will need to be used to create a link between the source and destination. Eventually the destination node will be out of range of the intermediate node, which will result in a second link breakage with no option to re-establish. This scenario is sufficient to test all the key features of AODV. The path loss model will be temporarily changed back to FreeSpacePathLoss to take advantage of the continuous 'good' connection over distance and the clear link break beyond maximum communication range, as indicated in Figure 3.6.

The simulation was run and communication between nodes was inspected. Route establishment, Hello messages and link breakage handling executed according to the information specified in Section 3.2.3.2. Route tables were also updated as expected. However, the route table fields did differ from Table 3.6. An extract from hostC's route table is shown in Table 3.8, indicating the routes to hostA and hostB. This extract was taken after hostC (IP 145.236.0.3) was the intermediate node providing an active route from hostA (IP 145.236.0.1) to hostB (IP 145.236.0.2). It can be seen that no hop count or next hop field is present in the route table. This, however, does not impact the actual functionality of the protocol. Therefore it is concluded that the protocol implementation is satisfactory.

Table 3.8: hostC route table extract

Entry	Value
[0]	dest:145.236.0.1 gw:145.236.0.1 mask:255.255.255.255 metric:1 if:wlan0(145.236.0.3) REMOTE AODV isActive = 1, hasValidDestNum = 1, destNum = 2, lifetime = 487.001464667128
[1]	dest:145.236.0.2 gw:145.236.0.2 mask:255.255.255.255 metric:1 if:wlan0(145.236.0.3) REMOTE AODV isActive = 1, hasValidDestNum = 1, destNum = 1, lifetime = 487.001464667128, precursor list: 145.236.0.1

3.2.4 Implementing Physical Obstructions

Obstructions present in the physical environment such as buildings will undoubtedly have a large negative effect on the performance of a VANET, especially in an urban environments. Obtaining a meaningful and trustable indication of how such a network could perform in the real-world, would require accurate modelling of buildings. There is of course much more to an urban environment than just buildings, but buildings are the primary LOS obstructions present in such an environment, therefore buildings will be focussed on

and other aspects such as vegetation, 3D terrain surfaces and detailed atmospheric influences will be disregarded. INET provides a compound module called PhysicalEnvironment with which one can model physical objects, or obstacles, as it will henceforth be named.

Obstacles are modelled in 3D space by making use of basic shapes with homogeneous material properties to provide an approximation of real-world scenarios. Loading obstacles into the simulation is performed on initialisation by defining object properties and positions in a XML configuration file. As mentioned in the INET documentation, adding a large amount of obstacles into the simulation will increase computational complexity when calculating signal propagation.

Unfortunately, no literature or documentation of the obstacle model validates the accuracy of its implementation, and blindly trusting the model without at least investigating the degree of effect it has on signal propagation and network performance would be unwise. To ensure that the presence of obstacles do indeed have the intuitively expected effect on signal propagation, a few small experimental simulations need to be performed before moving on to larger scale implementations.

The basis of this experimental simulation setup is the same as used in Section 3.2.2, thus results obtained in the aforementioned section can be used as a baseline for the case where no obstacles are present in the environment. To reiterate the scenario - a static source node will continually transmit UDP data packets to a mobile destination node at a constant data rate of 10 Mbps. The mobile node's initial position is identical to the source node, and it moves away from the source node in a straight line at a constant rate of 1 m/s, for a total time of 400 seconds. Radio parameters used are indicated in Table 3.2.

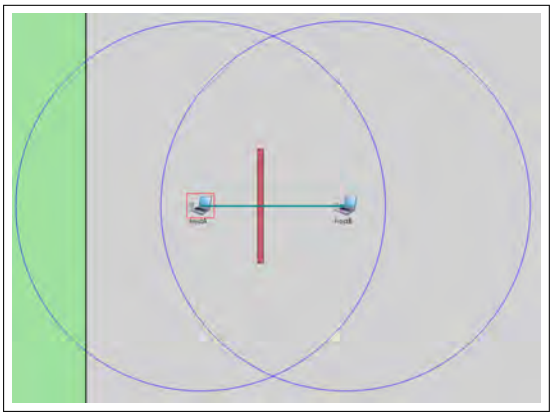
For the ground model, FlatGround was used. In INET, the ground model provides a height description (or 3D surface) of the physical terrain. There are two types of ground model to choose from at this time in INET: FlatGround, which is a flat surface at a specified height, and OsgEarthGround which can be used to define a 3D terrain surface. Modelling the physical terrain is outside of scope, thus FlatGround at an elevation of 0 m was chosen as the ground model and used throughout the rest of the project.

Furthermore, an obstacle loss model needs to be specified since the radio signals needs to interact with physical obstructions in the simulation. In INET, the obstacle loss model computes the power loss of a transmission based on its interaction with physical obstacles. Presently, there are two models available: IdealObstacleLoss and DielectricObstacleLoss. IdealObstacleLoss yields a total power loss when a signal penetrates an obstacle, or no power loss in a direct LOS scenario. DielectricObstacleLoss computes signal power loss based dielectric and reflection loss along the signal's path. Characteristics of physical objects are taken into account when the signal penetrates an object. This

model provides a much more accurate representation than IdealObstacleLoss of a signal's interaction with physical objects, and it is within scope to approximate power loss based on obstacles. DielectricObstacleLoss was chosen as the model to be used throughout the rest of the project whenever physical obstructions are involved.

A single obstacle will be placed along the path of the mobile node, which will block LOS between the nodes after a certain amount of time. During the initial stages of the simulation LOS will remain unobstructed until the mobile node moves through the obstacle, as indicated shown in Figure 3.8. Blue circles present in Figure 3.8 indicate the theoretical maximum communication distance of each node as calculated by INET. The obstacle is 100 m in length and 5 m in height, which is assumed to provide sufficient coverage such that all signals or reflections (if any) have to pass through the obstacle. Effects caused by adjustment of obstacle thickness, material properties and distance from the source will be investigated.

Figure 3.8: Geometry used to evaluate effects of material properties on signal propagation



Firstly, effects of material properties on signal propagation is isolated and investigated by keeping the obstacle at a fixed thickness and position. A thickness of 0.3 m was chosen to mimic a standard outer wall of a building at a distance of 50 m from the source node, whilst varying the material properties applied to the obstacle. INET provides various predefined homogeneous material properties, each having values specifying their resistivity, relative permittivity and relative permeability. 'Brick', 'concrete' and 'wood' were chosen as candidates for this experiment. FreeSpacePathLoss and NakagamiFading path loss models were compared as in Section 3.2.2, with results obtained from those simulations acting as the baseline.

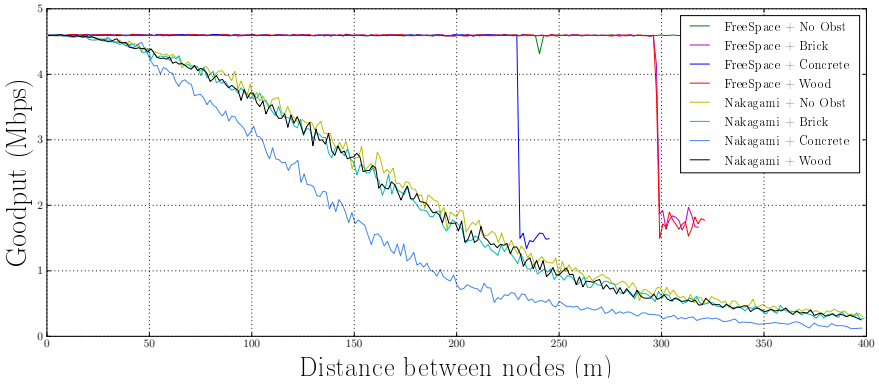


Figure 3.9: Goodput over distance with single obstacle of different materials

Figure 3.9 provides an overview of measured goodput over distance for various obstacle material properties, with a moving average window of 15 samples ($N = 15$) applied to the raw data. As seen in Figure 3.9, introduction of the obstacle negatively impacted goodput for FreeSpacePathLoss and NakagamiFading path loss models. 'Concrete' has the largest effect on performance, whilst 'brick' and 'wood' yielded similar results in the case of both path loss models. It is apparent in the case of NakagamiFading that 'brick' and 'wood' barely affected the performance when compared to the LOS scenario.

According to the simulation results a connection is still possible at 400 m, which is a counter-intuitive result. One would expect that a 0.3 m wall blocking LOS will have a larger impact on signal quality, especially a wall made of brick or concrete.

Strange anomalies are observed in cases where FreeSpacePathLoss is used - the goodput sharply drops to a lower value and cuts off at seemingly random times, depending on material properties used. This is not an accurate representation of what is expected in a real-world scenario, even more reason why FreeSpacePathLoss is not an optimal path loss model choice for this project.

Next, the impact of obstacle thickness and position will be investigated. In these experiments only 'concrete' and 'brick' materials were be used with FreeSpacePathLoss being eliminated due to results observed in Figure 3.9. Obstacle thickness of 5 m and 10 m was evaluated at distances of 20 m and 100 m from the source node, with results presented in Figure 3.10.

The first apparent observation is that only obstacle thickness has an effect on signal quality, positioning of the object does not seem to matter. This holds true for both concrete and brick materials as seen in Figure 3.10a and Figure 3.10b, where measured goodput at identical object positions converge to the same average result once the mobile node passed through the obstacle. Based on these results it can be argued that for this physical obstruction model, reduction of signal quality is not dependant on *where* in the signal's

path LOS breaks, merely *if* there is a break in LOS along that path. It can also be concluded that reduction in performance has a linear relationship to obstacle thickness, which can be attributed to material properties being applied homogeneously to the obstacle. Figure 3.10a provides a clear picture of this effect.

The brick material property still yields counter-intuitive results, since it is unthinkable that a signal of such low power in the 5 GHz range can pass so seamlessly through a 10 m brick obstacle. Concrete provides a better picture of what could be expected from a real-world scenario, with signal power completely lost when LOS is blocked by a 5 m or 10 m obstacle.

Implementation of obstacles in the simulation was successful, with the predefined concrete material being the most favourable choice for use in this project. It is still not known how well this model compares to empirical data, but with visual inspection and intuition it can be concluded from Figure 3.10b that this model would suffice the needs of this project for now.

3.3 Veins_INET and SUMO Maps

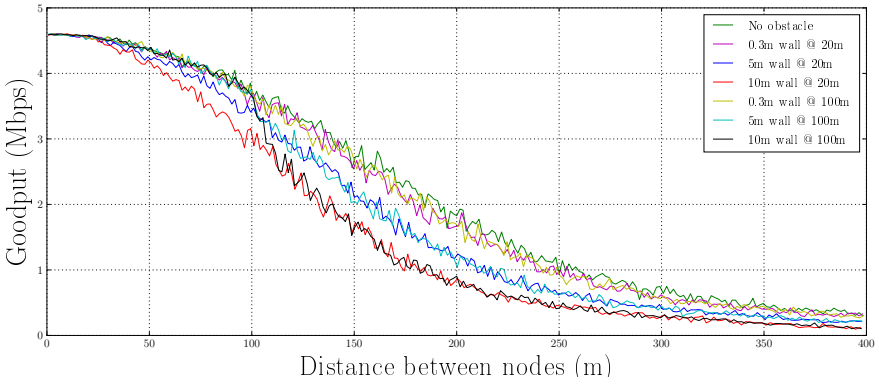
Veins_INET is a sub-project included in Veins framework which enables Veins to be used as the mobility model manager in INET, as described in Section 3.2. Basic functionality of Veins_INET was explained and tested in the aforementioned section, with the next step being to add this functionality to the network simulation as built up in the preceding sections.

3.3.1 Introducing Mobility

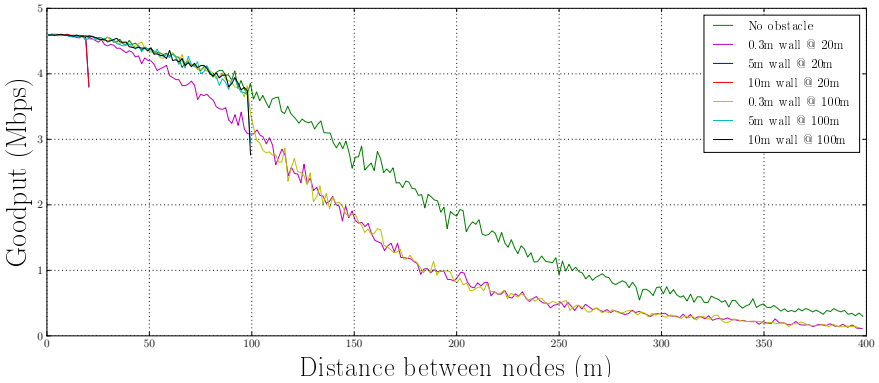
The hard-coded node definitions used up to this point are removed from the network description file, and the VeinsInetManager module is added which will manage node insertions and removals as well as their mobility characteristics. VeinsInetManager is a compound module from the Veins framework dedicated to create and manage vehicular network nodes in an OMNeT++ simulation using the INET framework. The VeinsInetManager requires a launch configuration file which contains paths to the SUMO map and route description files which are used to set up the SUMO simulation instance. For now the example configuration and SUMO files supplied by the sample project in Veins_INET will be used to get a basic simulation up and running.

VeinsInetManager also requires a network node module definition - all nodes added to the simulation by VeinsInetManager will be an instance of this module. An AODV-enabled wireless node module is created by extending AODVRouter (see Section 3.2.3.3) and adding HostAutoConfigurator. HostAutoConfigurator is an INET module responsible for network configuration, includ-

Figure 3.10: NakagamiFading vs single obstacle



(a) Brick obstacle with varying thickness and position



(b) Concrete obstacle with varying thickness and position

ing IP address assignment and routing table setup. It is deprecated and superseded by the IPv4NetworkConfigurator module, INET's most used and featureful network configuration module. Unfortunately, IPv4NetworkConfigurator does not function properly in cases where hosts are dynamically injected / destroyed in a simulation since it performs IP address assignment during simulation initialisation. Since the dynamic addition and removal of nodes / hosts are a requirement for this project, HostAutoConfigurator had to be used instead. Each host requires an instance of HostAutoConfigurator to perform dynamic IP address assignment. However, IPv4networkConfigurator is still needed in cases where a set of predefined nodes are present in the simulation, e.g. an aggregator or RSU. A static AODV-enabled aggregator node is added to the simulation (within communication range of at least one dynamic node throughout the simulation) to act as a data sink. Each dynamic node is also equipped with a UDP application layer that will periodically send data to the aggregator in order to test routing and communication of the newly imple-

mented dynamic nodes.

The simulation was run and all aspects discussed up to this point (node insertions / deletions, network configuration, routing etc.) was inspected and their expected functioning verified.

3.3.2 Creating a Custom SUMO Map

Simulating traffic in SUMO requires two key components - a file describing the road network and a file containing the start and end points for vehicles being simulated, called trips or routes. In its simplest form, a SUMO road network consists of nodes and edges. Nodes are descriptions of junctions or intersections whilst edges are descriptions of streets connecting these junctions. Routes describe the paths vehicles take whilst traversing the network, as well as characteristics of vehicles such as acceleration and maximum speed along these paths. Files containing this information are referenced in a **.sumo.cfg* configuration file which is used to set up and simulate a scenario in SUMO.

The area chosen for the urban networking evaluation for this project's VANET implementation is central Stellenbosch. Thus, a road network of central Stellenbosch is needed to generate and simulate traffic traversing the area.

3.3.2.1 Creating a Road Network

SUMO-road network files contain information that describe road networks and their attributes. Details such as street and road layouts, lane speed limits, lane directions, intersections and junctions as well as traffic light logistics are described in XML format. Such a file describing the road network of an area can be created manually for simple or hypothetical networks, but for more complex scenarios it is more efficient to use a third party source to obtain existing and up-to-date map data. One such source frequently used in the open-source community is OpenStreetMap (OSM). Data from the OSM database can be extracted to a file that can be used to create SUMO-road networks, since OSM freely provides all the road network information required in a SUMO-road network description. However, OSM files are not natively compatible with SUMO, can be converted by NETCONVERT to a SUMO compatible format. NETCONVERT, a utility created by the developers of SUMO, can be used to convert road networks from various sources to SUMO-compatible road networks and other optional formats.

Stellenbosch is a medium-sized town with a fairly dense urban core, laid out in a grid-like fashion. It is surrounded with low-density residential areas and vegetation, and has four main arterial roads providing access. No arterial roads pass through the central core, as seen in Appendix 6.3 Figure 2. Selecting

only the central part of town, will exclude most of the arterial road traffic from the SUMO simulation, shifting focus to a pure urban environment with little highway-like conditions. It is an ideal scenario, fitting the scope of this project, with the area of interest highlighted in Appendix A Figure 3 extracted from the OSM database on 22 April 2017. NETCONVERT is used to convert the OSM file to a SUMO-road network file. The method is documented in Appendix 6.3.

Now that a road network is generated, trips can be generated for the said network. This process is documented in Appendix 6.3.

3.3.2.2 Assessing the Road Network Map

After generating the road network and accompanying trips, a SUMO simulation instance was run to visually inspect the road network and vehicle driving patterns. A broad overview would suggest that the road network map is satisfactory, but various major flaws are identified upon closer inspection. Such flaws cause traffic to flow in unwanted and / or improbable areas.

Large areas of the map are affected by miss-classification of roads, non-traversable mountain trails, dirt paths and service roads are sometimes classified to be roads, whilst some roads are classified to be non-traversable. Figure 3.11 depicts the generated road network, with examples of areas affected by miss-classification indicated by red ovals. Black edges on this map represent roads that are traversable by vehicles added to the simulation, and light-grey edges represent non-traversable paths or roads. In these indicated areas it can be seen that the classification of roads is quite inaccurate, especially on the far left side of the map, which is known to consist of gravel paths and mountain trails and should not be accessible to vehicles in the simulation. This causes a substantial amount of unwanted traffic to traverse these roads, as indicated in Figure 3.12 (a). Examples of other flaws such as disconnected roads are indicated by red circles. This might seem like a minor issue, but depending on the location of the disconnection the impact could be severe. As indicated by the small circle in the top-left corner, a major road suffers a disconnection which resulted in no vehicles traversing this section of road at all.

Another issue present in the map has to do with its physical layout and not the conversion algorithm. Large sections of the road network are outside the urban core, indicated by the blue rectangles in Figure 3.11, and it is not desired for vehicles to traverse these areas. Among these areas are low-density residential zones which is outside the scope of this project. The many discontinuities and unsuitable areas derail the main objective of having the most traffic in the urban core of the map. As seen in Figure 3.12 (b), an extract from within

the blue rectangle in the top-right of the map, many vehicles (represented by yellow triangles) spend time in this specific residential zone.

Figure 3.11: Generated road network map of central Stellenbosch

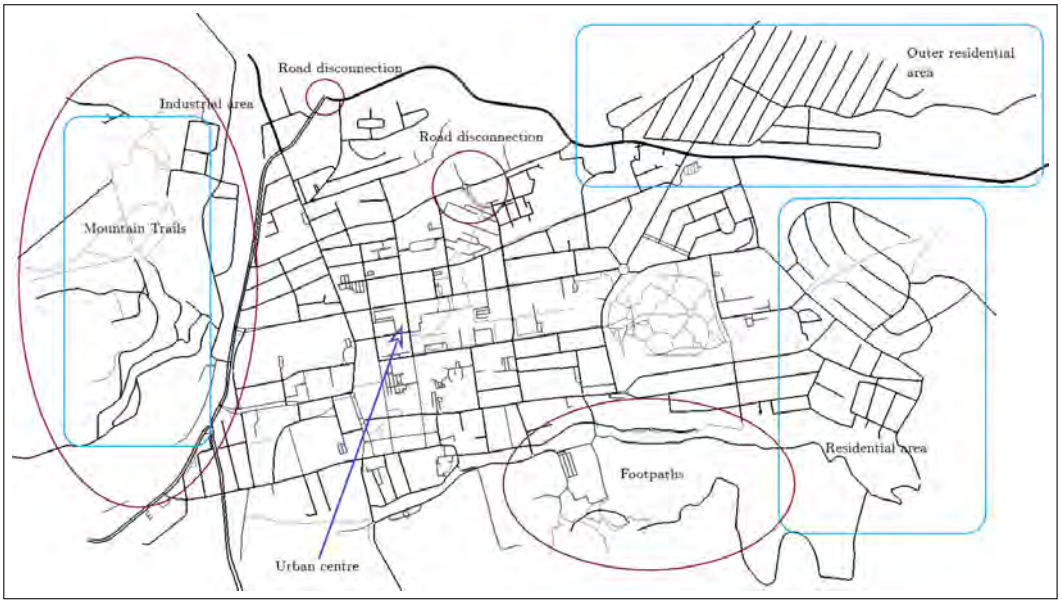
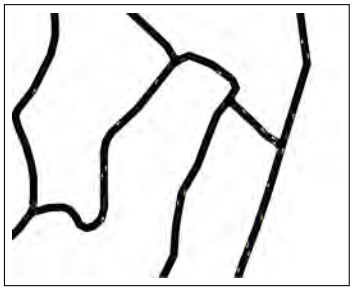
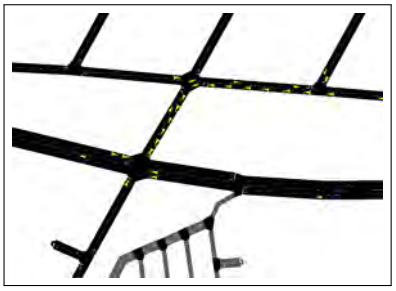


Figure 3.12: Vehicles (represented by yellow triangles) in unwanted locations



(a) Mountain trails



(b) Residential area

3.3.2.3 Fine-tuning the Road Network Map

At present it is unknown if the above mentioned road classification issue is due to flaws in NETCONVERT's algorithm or a result of possible ambiguity in OSM's description of the map itself. Taking a closer look at the map data downloaded from OSM's database may provide some insight. Java OpenStreetMap Editor (JOSM) is a free application which allows for offline viewing and editing of OSM maps. Offline editing is preferred in this case since any alterations made to the map would be specific to this project, and these changes

should not be pushed to the OSM database. The downloaded map was opened with JOSM and presented in Appendix 6.3 Figure 3.

All features in an OSM map are described key-value pairs called *tags*. Tags are attached to basic data types such as nodes (single points in space) and ways (list of nodes) which in turn make up larger structures such as buildings and roads. Any path that allows travel in some form, albeit vehicular, pedestrian or cycling, is described by the *highway* tag. Values attached to this tag will classify the path based on its importance or priority, e.g. *motorway* value will indicate that a road is a major divided highway whilst the *tertiary* value will be applied to roads that link smaller towns and villages.

Inspection of the tags in the affected areas indicated in Figure 3.11, yields some interesting insight. It becomes apparent that NETCONVERT makes use of tags to classify roads based on the value of the tag, since ways with *highway* keys and values such as *service*, *footway* and *path* are all classified to be non-traversable whilst values such as *tertiary*, *residential* or *unclassified* will result in roads being traversable.

It is concluded that flaws in the SUMO-road network map is caused by two factors: 1) Paths carrying the *highway:track* tag are classified as traversable roads, which is not optimal in this case. 2) Certain sections of roads have the wrong tag causing them to be non-traversable, e.g. some roads that are not service roads carry the *highway:service* tag. These issues are fixable by simply updating all tags to reflect desired values. All *track* values were replaced with *path*, and all road sections containing incorrect values were assigned the value of the road that they are connected to.

An identical approach can be used to get rid of unwanted roads in the blue areas highlighted in Figure 3.11 without having to delete nodes manually in JOSM. This brought up a question though - surely NETCONVERT has to provide some sort of functionality to deal with such scenarios? It is conceivable that this might be a common dilemma faced by users creating their own SUMO road networks from OSM data. At this point a more in-depth study of NETCONVERT's documentation was performed, and it was seen that typemaps might aid in such a solution.

In the context of NETCONVERT, a typemap is a file which contains attributes or restrictions that are applied to certain road types when converting an OSM map. Other than this, no more information on typemaps is supplied in the documentation, which is why it was originally overlooked. After browsing through typemaps contained in *<SUMO_BASE_DIR>/data/typemap*, it was deduced that these files also determine which specific road types should be imported by NETCONVERT since OSM tags present in the default typemap file were the same tags identified earlier when debugging the map with JOSM. This information correlates well with the observations made previously in this

section.

Now a custom typemap was created containing only the road types required in this scenario. By omitting tags such as *highway:residential* and *highway:pedestrian* and changing the tags of unwanted sections of road to one of these types, a more ideal SUMO-road network was created to suit the project's needs. All roads outside the urban core was omitted, but main roads that feed traffic in and out of the area was kept. The result of these adjustments can be seen in Appendix 6.3 Figure 4.

3.3.3 Converting OSM Buildings to INET Obstacles

Obstacles representing buildings in the network simulation is just as important as the SUMO-road network map and vehicle routes when attempting to mimic a real-world scenario of a vehicular network in an urban environment.

3.3.3.1 Importing OSM Map Features

OSM maps does not only contain layouts of road networks and points of interest, but also additional features such as various building types, parks, forests, etc. These features are provided in the form of polygons with associated tags for classification. It is desired to convert this polygon data to INET obstacles and position them accordingly in the network simulation.

SUMO is capable of visualising various map features defined as polygons in a XML shape file. These features have no effect on the traffic simulation itself - it is merely a visual addition to the SUMO simulation UI to aid in debugging and enhancing its appearance. The interesting part of this is that OSM map features can be converted to a SUMO-compatible format by using a utility called POLYCONVERT, which will result in correctly positioned polygons in relation to the SUMO road network map. Information contained in such a shape file can possibly be used to create the required obstacles for the network simulation.

Just like NETCONVERT, POLYCONVERT can use a typefile to determine which features to import and to specify their attributes. In this case, only features that represent buildings need to be imported - they are represented by the *building* and *amenity* tags in OSM maps. Figure 4 in Appendix 6.3 presents the refined road network with all features hidden except *building* (pink) and *amenity* (yellow) tags. It can be seen that amenity tags have to be added since some buildings in the central Stellenbosch area are classified as amenities. However, school ground borders are also classified to be amenities, and not only school buildings, as indicated by the large yellow areas in Figure 4. Such areas were given the *grass* tag instead. A shape file was created by using

POLYCONVERT as indicated in Appendix 6.3. When the *–net-file* parameter is used (as shown in the Appendix), POLYCONVERT will project the coordinates of the extracted features to align perfectly with the specified road network as in the original OSM map.

3.3.3.2 Poly to Obstacle Conversion

Obstacles in INET are created with basic shapes, as mentioned in Section 3.2.4. Polygon definitions contained in the XML shape file have to be interpreted and converted to INET compatible shapes.

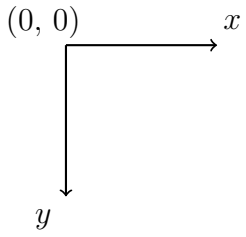
POLYCONVERT projects the features imported from OSM onto the first quadrant of a 2D Cartesian plane in meters. An example of such a definition describing a rectangular shape is displayed below.

```
<poly id="429853261" type="building" color="255,230,230" fill="1"
layer="-1.00" shape="623.32,1173.10 634.62,1175.69 636.74,1166.53
625.44,1163.94 623.32,1173.10"/>
```

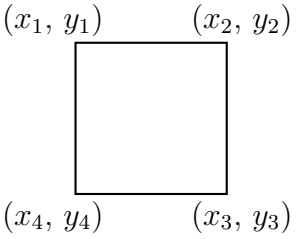
The only parameter of interest is the list of 2D coordinates provided by the *shape* attribute. These coordinates describe the feature's shape, and inherently its location, by following axis conventions normally used computer by graphics. When using this method of describing the Cartesian plane, the origin is placed in the top-left corner and Y coordinates increment as you proceed downward away from the origin. An example of such an coordinate system is displayed in Figure 3.13a. To illustrate how shapes are described when using this method, the simple polygon in Figure 3.13b will have a *shape* attribute of the following:

$$shape = (x_1,\, y_1\; x_2,\, y_2\; x_3,\, y_3\; x_4,\, y_4\; x_1,\, y_1)$$

Figure 3.13: Shape description example



(a) 2D Cartesian axis in computer graphics



(b) Four-edge polygon

INET has a similar method of describing shapes, with prism and polyhedron shape types used to create irregular polygons. Prisms have a 2D polygon as

a base, and are extruded vertically to a specified height. Polyhedrons in turn need to be described in 3D space with a full set of coordinates. In INET the physical simulation environment is also described with a Cartesian coordinate system (in meters), with the origin being in the top-left corner, which results in no projection or transformation being required to plot a SUMO shape in the same position in INET's simulation space. The simplest method of creating obstacles in this case would be to use prisms, since the way of describing prisms in INET is very similar to that of polygons in SUMO. Given this information, the polygon depicted earlier can be described as a prism obstacle (10 m height) in the following manner,

```
<object position="center x_center y_center 0" orientation="0 0 0"
shape="prism 10 <2D Coords List>" material="concrete"
fill-color="203 65 84" opacity="0.3"/>
```

with *<2D Coords List>* extrapolated to be:

$$(x_1\ y_1\ x_4\ y_4\ x_3\ y_3\ x_2\ y_2)$$

The mandatory *position* parameter specifies the anchor point of the shape, which was chosen to be the center in this case. Thus, *x_ center* and *y_ center* would represent the center coordinates of the shape's bounding box.

A Python script was created to convert all polygons defined in the XML shape file to a PhysicalEnvironment configuration file. A few key assumptions and decisions were made during this step:

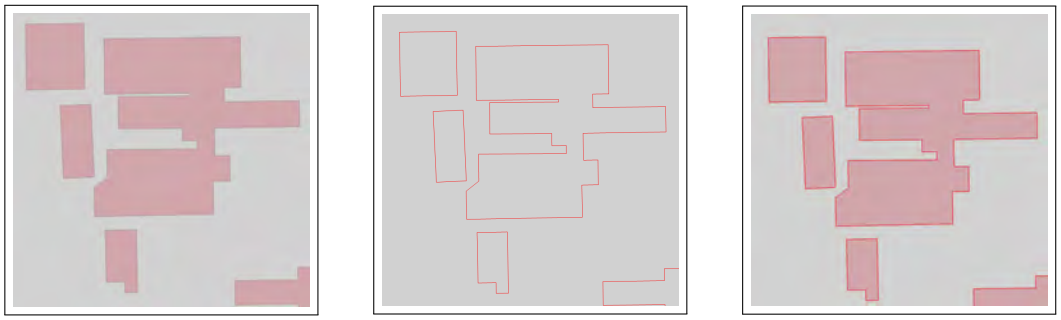
- Implementing buildings of various heights to obtain a more accurate model of the central Stellenbosch area would have a negligible affect on overall network performance. All buildings are assumed to be of the same height - 10 m.
- All buildings are assumed to be solid, homogeneous obstacles.
- All buildings will be given identical material properties.

3.3.3.3 Verifying Obstacle Placement

It is important that the obstacles in the INET simulation space are aligned correctly with the SUMO road network. Veins framework (not to be confused with Veins_INET) has modules called ObstacleControl and AnnotationManager responsible for instantiating and displaying its own obstacle models in OMNeT++. Since it is designed to be integrated with SUMO, these modules make use of XML shape files to create obstacles which are correctly aligned with the road network. It is possible to use these modules in an INET simulation to draw an overlay of the obstacles as interpreted by Veins. This overlay

can be used to verify the placement of INET obstacles generated previously. An example of this can be seen in Figure 3.14.

Figure 3.14: Using Veins to verify obstacle alignment



(a) INET obstacles (b) Veins obstacles (c) Obstacle overlay

This approach was verified by observing and comparing vehicle positioning in relation to the obstacles to that of a standalone SUMO simulation with the same configuration files. It was determined that obstacles are indeed positioned and aligned correctly as desired.

3.4 Summary

In this chapter, a network simulation environment implementing IEEE 802.11p, AODV and path loss models was integrated with a vehicle traffic simulation environment which simulates traffic on a road network map of central Stellenbosch. Physical obstructions were added to the network simulation in correct relative positioning to the road network map. This simulation environment is capable of evaluating IEEE 802.11p and AODV for suitability of realising the envisaged solution, but needs to be verified and calibrated to ensure it provides a realistic approximation of expected real-world results.

Chapter 4

Hardware Solution and Simulation Verification

Evaluating the viability of implementing a distributed licence plate detection system with VANET-specific standards and protocols is the main goal of this project. Performing real-world experimentation would provide empirical data to support and aid in answering the questions raised in Chapter 1.

An ideal method of evaluating the standards and protocols as well as the implementation of the envisaged system would be to install reputable commercially available VANET OBUs and imaging hardware in a large number of vehicles, and gather data over the period of a few months. Evaluation of such an implementation would provide a clear picture of how such a system would function in an ideal real-world VANET scenario with no assumptions or approximations.

Due to budget constraints discussed in Chapter 2 it is not possible to equip many vehicles with commercial or custom, affordable hardware solutions in order to perform the required experimentation. As mentioned in Chapter 2, it was chosen to use network and traffic simulation to evaluate the standards and protocols for the proposed system implementation. Even though INET framework and its models are trusted by many researchers to provide a solid and reliable approximation of real-world networking, there are almost no publications performing real-world experimental verification of the models and components of interest in this project. Having access to empirical data in order to support and validate results obtained from the networking simulation would be of great aid in order to create a trustworthy simulation environment. Empirical data can also be used to optimise the simulation setup providing a better approximation of the real-world scenario.

It is assumed that using real-world hardware in parallel with network simulation would be beneficial in order to:

- Verify implementation of networking models such as signal fading, obstacles and the routing protocol
- Use empirical data to improve and fine-tune the network simulation parameters
- Use hardware as a tool to investigate and gain insight into possible problems that may be faced in the network simulation
- Gain a better understanding of complications that may arise during real-world implementation of such a system
- Obtain a better frame of reference with which to answer the research questions

Hardware is not the main focus of this project, but it does play a key role in successfully answering the research questions at hand. The main point of performing hardware experimentation in this case is to *complement* and *improve* the network simulation, and to be a useful *tool* which can be used to gain understanding and insight into evaluating the standards and implementing the envisaged system.

4.1 Selection of Hardware

Evaluating aspects such as signal fading or the MAC implementation will require only two network nodes, but in order to evaluate multi-hop communication a minimum of three nodes are needed. Due to limitations set forth by the allotted budget for this project, it was decided to settle on a maximum of five nodes which at least provides more flexibility than having merely three. Ideally one would like to have as many nodes as possible which would allow for experimentation with a large variety of possible scenarios.

Each node requires three hardware components: A wireless network adapter, a compute platform and external antenna(s) as mentioned in Section 2.3.2. Selection of the wireless adapter and compute platform is dictated by requirements set forth in the aforementioned section. It is critical that each node has identical hardware in order to eliminate as many variables as possible during the experimentation process.

4.1.1 Wireless Adapter

The wireless adapter needs to contain one of the Atheros chipsets listed in Table 2.7 and have a mini-PCI or mini-PCIe interface. Local availability of qualifying adapters was basically non-existent, and came at a hefty price. Modules

were obtained from a reputable online retailer such as Amazon, since they had a large variety of wireless adapters with AR9462, AR9280 or AR9382 chipsets. The Hewlett-Packard AR5BHB92-H 802.11abgn Wireless Half-size Mini-PCIe Card was selected as it has an AR9280 chipset. This choice was made due to the reputability of the manufacturer (the card is an OEM module used in many HP laptops) and the product having a multitude of positive reviews on Amazon.

4.1.2 Compute Platform

As discussed in Section 2.3.2, the processing platform has no major specification constraints. It requires a mini-PCIe slot, as per wireless adapter selection, and compatible with Linux with kernel version 2.2.27 or later.

Laptops, embedded computers, and all-in-one PCs could be compatible with this role, depending on the hardware configuration. It would however be beneficial to obtain devices with a small footprint and low power consumption so that portability and installation of devices in vehicles would not become an additional roadblock. A company called Telectro CC offered the use of five Proline NT425 single-board computers for use during this project at no cost. These systems have mini-PCIe slots and are capable of running traditional 32-bit desktop distributions of the Linux operating system. Comparison of various compute platforms is excluded from this document, as the single-board computers used in this project was obtained free and met the necessary requirements.

Proline NT425 is a small form factor single-board desktop computer with specifications indicated in Table 4.1. It requires a 19V DC input supplied by an external power adapter as used by many laptops. This is the only major drawback of this platform, since a custom power adapter or power inverter would be needed when such a system is installed in a vehicle.

4.1.3 Antenna

Miniature RF connectors, also called U.FL or IPX connectors, are present on the mini-PCIe wireless card which are used for connecting external antennas. Most embedded systems make use of omnidirectional Printed Circuit Board (PCB) antennas attached to their wireless adapters due to the extremely small footprint and low cost.

Size, cost, efficiency, bandwidth and radiation pattern are all important factors when selecting an appropriate antenna. It should be noted that the antenna choice was of importance as it was observed that the antenna plays an integral role in the performance and characteristic of the system.

Initially, a 4 dBi omnidirectional PCB antenna was selected mainly due to its small footprint, which meant that it could be placed inside the single-board computer's casing. However, it was observed that the nodes did not have the expected communication range. A 2 dBi omnidirectional whip antenna, similar to those commonly found on wireless router devices in personal home networks, was chosen to replace the PCB antenna. This antenna had better performance in terms of power output and efficiency in all directions. The whip antenna was used during all tests unless otherwise specified.

4.1.4 Summary

Wireless adapters, single-board computers and omnidirectional antennas were obtained with which five identical test platforms (or nodes) could be created. The hardware specification of the final testing platform is available in Table 4.1, which was used during all real-world experimentation and testing throughout this project.

Table 4.1: Final hardware components specification

Hardware Components Specification			
Wireless Network Adapter		**Proline QBox NT425 Single Board Computer**	
Model Name	HP AR5BHB92-H 802.11abgn Wireless Half-size Mini-PCIe Card	**CPU**	Intel Atom D425 1.8 GHz
FCC-ID	PPD-AR5BHB92-H	**RAM**	2 GB DDR2, 800 Mhz
Chipset	AR9280	**HDD**	320 GB, 5400 RPM
Output Power at 5 GHz	20.41 dBm (109.90 mW) combined average without antenna	**Power Supply**	19V DC, 3.42A
Antenna	One 2 dBi omnidirectional whip antenna	**Dimensions (mm)**	135 x 190 x 25

4.2 Software Setup

The selection and combination of hardware is only the first part of of constructing a VANET node. Keep in mind that none of the hardware components used in this project are tailor made to function in an ITS scenario, they are commercial products intended for office or home usage. Software, correct implementation thereof, is a critical component which will enable the hardware to function within the desired specification.

To have a fully functional IEEE 802.11p compliant device capable of multihop data transfer, requires MAC and PHY layers to support IEEE 802.11p, and a routing protocol (AODV in this case) needs to be implemented.

4.2.1 Overview of IEEE 802.11p in Linux

It was explained in Section 2.3.2 that Linux will be used as the operating system, due to the requirement of modifying the ath9k driver. However, to

fully implement IEEE 802.11p in Linux, much more than just a small driver modification is needed [70].

To understand why changes need to be made to a 'vanilla' Linux installation and how such changes will enable implementation of IEEE 802.11p on the PHY and MAC layers, a brief overview of IEEE 802.11 wireless networking in Linux is required. IEEE 802.11 implementation in the Linux kernel-space consists of the following main components: *mac80211*, *cfg80211*, *nl80211* and drivers such as ath9k [88].

mac80211. As mentioned in [88], most modern 802.11 wireless Network Interface Card (NIC)s (WNIC) (including devices using the ath9k driver) do not implement the MAC layer in hardware, but rather relies on drivers to implement and manage the MAC layer. WNICs whose MAC Layer Management Entity (MLME) is managed in software is known as *SoftMAC* devices. *mac80211* is the framework which is used implement drivers for SoftMAC WNICs - it is essentially a driver API.

cfg80211 acts as a bridge between the userspace and *mac80211*, providing a configuration API which is used to configure 802.11 devices and drivers. *cfg80211* also aids in enforcing regulatory spectrum compliance.

nl80211 enables inter-process communication between userspace processes (such as configuration utilities) and the *cfg80211* kernel subsystem by making use of Netlink sockets.

Czech Technical University launched a project on Github called *802.11p-linux* [89] which aimed to develop a IEEE 802.11p implementation for Linux, with the end goal of being a stepping stone to a fully integrated V2V networking protocol stack. They developed a fully functional implementation, which was included in the mainline Linux kernel version 3.19 [88].

Modifications were made to the IEEE 802.11 wireless networking subsystem mentioned earlier, as well as changes to the ath9k driver. The most important changes will now be discussed.

In *mac80211*, support for OCB mode is added, BSSID is set to a fixed value of 48 '1' bits when in OCB mode and EDCA parameters are adjusted to match the IEEE 802.11p specification.

cfg80211 was changed such that it can configure a wireless interface to communicate in OCB mode. An OCB network can virtually be joined by specifying a center frequency and bandwidth, and left, by disabling RX/TX.

Additional commands were added to *nl80211* with which to configure OCB mode via userspace utilities such as *iw*.

In the ath9k driver, operation in the 5.9 GHz band and utilisation of channels with 10 MHz bandwidth is enabled which allows the device to tune in to channels in 5850-5925 MHz range. Functionality to join and leave an OCB network was also added.

4.2.2 Setup of 802.11p-enabled Linux

With the prerequisites for a fully functional IEEE 802.11p Linux environment known, a distribution and kernel version needs to be selected.

4.2.2.1 Selection of Linux Distribution and Kernel Version

Due to the processing platform being a consumer desktop product with relatively powerful specifications compared to many embedded computers, Ubuntu was chosen as the distribution of choice. This distribution has great community support and a large user base which could prove useful when odd problems arise. It is also guaranteed that Lisov *et al.* [88]'s amendments will be included in any version of this distribution using kernel version 3.19 or later due to it using the mainline Linux kernel.

Ubuntu 15.04 Vivid Vervet using kernel version 3.19.8 was selected as the operating system since it was the first Ubuntu release to implement the modified 802.11 subsystem, and it is known that the modified ath9k drivers provided by the *802.11p-linux* project[1] will be compatible with this kernel version.

The specified operating system was installed on all five nodes, with repository lists updated due to Ubuntu 15.04 being End of Life (EOL) and not officially supported any more. Repositories indicated in Appendix 6.3 Figure 9 was added to the `/etc/apt/sources.list` file.

4.2.2.2 Compiling and Installing a Modified Kernel

Not all additions proposed by the *802.11p-linux* project were merged into the mainline kernel, and this includes the modified ath9k drivers. Updating the ath9k driver requires modification of the kernel source files followed by configuring and compiling the modified version of the kernel.

Obtaining the source files for the currently installed kernel and required packages to build a kernel is done by executing:

```
$ apt-get source linux-image-$(uname -r)
$ sudo apt-get build-dep linux-image-$(uname -r)
$ sudo apt-get install build-essential libncurses5-dev gcc libssl-dev
```

[1] https://github.com/CTU-IIG/802.11p-linux/tree/its-g5_v3/drivers/net/wireless/ath/ath9k

`grub2 bc`

The ath9k driver located in `/drivers/net/wireless/ath/ath9k` can then be replaced with the driver files from the *802.11p-linux* project, and the regulatory rules in `../ath/regd.c` need to be updated.

Now the kernel needs to be configured in order to enable OCB mode, by executing `$ make menuconfig` and enabling `MAC80211_OCB_DEBUG`, `CONFIG_MAC80211_STA_DEBUG`, and `ATH9K` as shown in Appendix 6.3 Figure 10 and Figure 11 respectively.

The kernel can be compiled by calling `$ make`, and installed after compilation with `$ sudo make modules_install` followed by `$ sudo make install`.

After installation, the bootloader needs to be updated by editing the `/etc/default/grub` file. All `GRUB_HIDDEN` definitions should be removed, and the word *splash* should be removed from `GRUB_CMDLINE_LINUX_DEFAULT`. The rest of the definitions should match those displayed in Appendix 6.3 Figure 12. To apply the changes, `$ sudo update-grub` is called. The system can now be rebooted to the new kernel.

Installing the new kernel on other nodes can be simplified by creating installable `.deb` packages with `$ make deb-pkg`. This will result in five packages which can be copied and installed on other nodes by executing `$ dpkg -i linux-*.deb` in the directory containing the packages.

4.2.2.3 Configuring OCB Mode

After installing the new kernel, the wireless configuration utility (`iw`) and regulatory information needs to be updated too before a wireless interface can be configured to operate in OCB mode. A developer of the *802.11p-linux* project created a helpful guide on setting up the required utilities and regulatory information [90], thus the process of setting up these aspects are not repeated in this document.

Netlink development library and headers are used during this setup (as well as in the next section), and there is a bug in this version of Ubuntu, which causes these packages to be installed in an unexpected directory. This will result in the packages not being found or detected during compilation of code dependant on these libraries and headers. Creating a symbolic link to the expected directory, as shown below, will solve this issue:

```
$ sudo ln -s /usr/include/libnl3/netlink /usr/include/netlink
```

After installation and configuration of the required utilities and regulatory information, the wireless interface (in this case `wlan0`) can be configured to operate in OCB mode as follows:

Figure 4.1: Configuring wireless interface to operate in OCB mode

```
$ sudo iw reg set DE
$ sudo ip link set wlan0 down
$ sudo iw dev wlan0 set type ocb
$ sudo ip link set wlan0 up
$ sudo iw dev wlan0 ocb join 5860 10MHZ
$ sudo ip addr add 10.2.1.1/24 dev wlan0
```

The regulatory zone is set to `DE` as seen in Figure 4.1, the region where the developers of *802.11p-linux* reside. `[ (5850 - 5925 @ 20), (100 mW), NO-CCK, OCB-ONLY ]` is the entry which was added to comply with ITS-G5 spectrum regulations in Germany, which is sufficient for this project as well. It is however possible to add and modify custom regulatory rules if required during the extent of this project.

ITS-G5 SCH4 (or WAVE channel 172) was the selected channel of operation as seen in Figure 4.1, to match that of the network simulation as discussed in Section 3.2.1. To avoid ambiguity, each node was assigned a static IP address in the `10.2.1.x` range to act as a constant unique identifier during experimentation and result analysis.

After setup and configuration, successful single-hop ad-hoc communication between nodes was confirmed by placing nodes close to each other and using the Linux `ping` command. `ping` triggers the ICMP protocol's `ECHO_RESPONSE` by sending `ECHO_REQUEST` datagrams to a host and logs the time it takes for responses to arrive. Network connectivity indication, packet loss rate and latency can be obtained by using this utility.

Further verification and experimentation is performed and discussed in Section 4.3.

4.2.3 AODV Implementation

Due to the popularity of AODV, one would assume that a large amount of up-to-date open-source implementations are available for use in Linux - this is however not the case.

4.2.3.1 Availability of Source Files and Implementation Selection

AODV implementations developed for the Linux operating system include AODV-UU [91], AODV-UCSB [92], AODV-UIUC [93] and Kernel AODV, which are all discussed in part by Höfner *et al.* [94]. None of these implementations are recently modified or up-to-date, and source files for only AODV-UU and AODV-UIUC could be found. It would not be viable to discuss and compare these different implementations, since only AODV-UU or AODV-UIUC could realistically be used in this scenario.

Even though source files for AODV-UU and AODV-UIUC are available, compatibility with recent Linux kernel versions is still an issue. AODV-UIUC was developed for kernel version 2.4 and last updated in 2002, whilst AODV-UU was last updated in 2011 and is compatible with kernel version 2.6. AODV-UU was chosen to be the AODV implementation of choice due to it being compatible with a more recent kernel version, and the fact that it has a widely used NS-2 and NS-3 implementation adds trustworthiness to its validity.

4.2.3.2 Brief Overview and Installation of AODV-UU on Kernel 2.6.x

AODV-UU was developed by Uppsala University for use in their ad hoc protocol evaluation testbed, hence the 'UU' extension in the name. According to the project documentation [91], AODV-UU is based on their interpretation if RFC 3561 with stability being the focus point rather than raw performance.

In short, the AODV routing protocol itself is implemented in user-space as a daemon which communicates with kernel modules via Netlink in order to update the route table by using Netfilter. Packets are captured by Netfilter, but this filtering is performed in the user-space.

Authors guarantee the implementation will perform as expected with up to four hops in an ad hoc network, however larger networks were not tested. Since a maximum of only five nodes will be used in real-world testing during this project, this claim is promising.

To investigate the functionality of AODV-UU, Ubuntu 10.04 LTS with kernel version 2.6.32 was installed on three nodes and AODV-UU was successfully compiled and installed. All nodes need to be set to ad hoc mode and assigned a unique IP address - this process is shown in Figure 4.2.

AODV-UU is then run as a user-space process in the console by calling `$ sudo aodvd -r 5`, which will also log the route table content every 5 seconds. The process can be run as a daemon by passing the *-d* parameter, and local link repair can be enabled with the *-L* parameter. In this case, local link repair was not enabled due to it not being implemented in the network simulation.

Figure 4.2: Ad hoc mode setup procedure

```
$ sudo service network-manager stop
$ sudo ifdown wlan0

$ sudo iwconfig wlan0 mode ad-hoc
$ sudo iwconfig wlan0 channel 4
$ sudo iwconfig wlan0 essid vanet

$ sudo ifconfig wlan0 up
$ sudo ip addr add 10.2.1.1/24 dev wlan0
```

4.2.3.3 Updating AODV-UU to Kernel Version 3.19

Linux kernel 3.19.8 is used in this project whilst AODV-UU is designed to work with kernel versions 2.4.x and 2.6.x, thus it is required to modify the AODV-UU source files to enable installation and compatibility with the newer kernel.

As mentioned earlier, protocol logic runs in a userspace daemon and Netlink sockets are used to communicate with kernel modules. Changes would need to be made to the Netlink code as well as some parts of the kernel modules, since kernel function definitions may have changed in newer versions of the kernel.

Performing an in-depth study of the AODV-UU source code and comparing the function definitions and usage across kernel versions would be extremely time consuming. An approach of iterative adjustments to the source code was taken after briefly studying the the kernel module code and implementation of Netlink communication to gain a basic understanding of the functionality.

Firstly, the makefile was adjusted to enable compilation on the newer kernel version. After executing the makefile, errors encountered during the build process were investigated and resolved. Such errors were caused by function definitions that have changed or do not exist any more, and could be solved by updating function calls or by replacing obsolete functions with newer versions that serves the same or similar purpose. Other parts of the code also had to be changed to facilitate the use of these new functions.

Once the source files could be built without errors, the modules were installed and code-related issues encountered during the installation were resolved in the same manner.

Successful compilation and installation does not mean the implementation will function correctly or as intended. To verify the functionality, AODV-UU was installed on two nodes running Ubuntu 10.04 LTS with kernel version 2.6.32, whilst Ubuntu 15.04 and the modified AODV-UU was installed on a third

node. Various aspects such as link establishment, link breakage, multi-hop data transfer and route timeout were tested and compared across implementations. Issues such as routes not being added or updated was attributed to occasional Netlink message transfer failure to the kernel-space. Bugs were tracked down and resolved by making use of verbose debugging, until the implementations functioned the same.

It should be noted that no changes were made to the actual routing algorithm running in userspace, thus the updated version of AODV-UU retains interoperability with the original even though they are running on different kernel versions. Changes were purely kernel-space and Netlink based.

4.2.3.4 Contribution to the Open-Source Community

Although AODV-UU was modified with the intention to specifically aid this project, it was seen as an opportunity to contribute to the open-source community.

The source code was adjusted in such a manner to make this implementation of AODV-UU compatible not only with kernel 3.19, but also all kernel versions between 2.6 and 3.19.8. Due to AODV being a popular protocol in research, it is envisioned that this part of the project could provide a useful, albeit small, contribution to the open-source community and networking research. Researchers interested in implementing AODV in more recent kernel versions could hopefully benefit by this revised implementation.

This adjusted AODV-UU version is publicly available in the form of a GitHub repository[2] called “aodv-uu-for-kernel-version-3.19" . Although the implementation was extensively tested on kernel version 3.19.8 and all visible issues were resolved, it is not guaranteed or intended to be an error-free solution for use in critical applications. Also note that extensive testing was performed *only* with kernel versions 3.19.0 and 3.19.8.

4.3 Hardware Experimentation

Hardware nodes now have a functional implementation of IEEE 802.11p as well as AODV, matching the network simulation’s implementation. Experimentation with these nodes in various scenarios will be used as a tool to obtain empirical data of the actual characteristics and performance metrics of such an implementation in the ‘real-world’. These results will be used to calibrate the network simulation accordingly, such that the network simulation will be

[2] https://github.com/ctruter/aodv-uu-for-kernel-version-3.19

able to provide a more accurate representation of real-world performance.

Various key aspects such as throughput in ideal LOS scenarios and signal fading in open space will be investigated in the following section. Data will be compared by making use of specific performance metrics. The rest of this section is presented as follows: Explanation of overall testing methodology used during experimentation, explanation and description of performance metrics used, testing of direct link communication (single hop, no routing protocol implemented) in LOS scenarios, investigation of the impact of obstacles in NLOS scenarios, investigation of routing protocol in single- and multi-hop scenarios and finally discussions and conclusions. Each of subsections regarding experiments will briefly describe the experimental setup and methodology used, followed by presentation and interpretation of obtained results, as well as comparison to identical scenarios in the network simulation.

4.3.1 Direct Link Communication

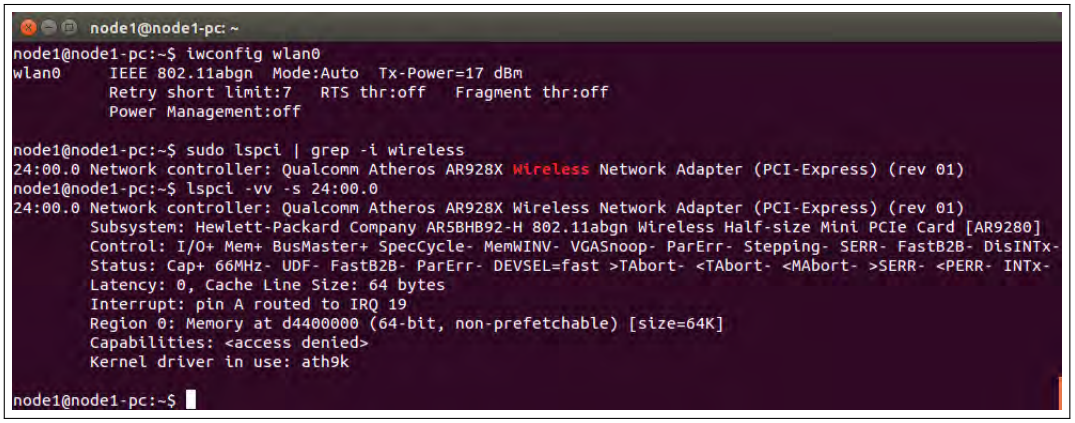
Getting a baseline of performance metrics such as maximum achievable throughput and latency in ideal conditions is a good place to start when comparing simulation to reality in this case. It should be mentioned that wireless adapter parameters and settings (e.g. transmission power, bit rate) will remain untouched throughout this project to get a sense of default, 'out-of-the-box' performance.

4.3.1.1 Obtaining Default Configuration

Firstly, these default parameters should be identified. To obtain information about network interface configuration in Linux, tools such as `iwconfig`, `ifconfig` and `wavemon` can be used. `ifconfig` and `iwconfig` is similar in the sense that they can both be used to configure and view information about network interfaces, but `iwconfig` is dedicated to wireless interfaces. Figure 4.3 displays the default interface parameters with `iwconfig` as well as using `lspci` to display detailed information of the wireless device (WNIC) itself. Note that this is the default configuration *after* enabling OCB mode as described in Figure 4.1.

It can be seen in Figure 4.3 that the WNIC in use is indeed the HP AR5BHB92-H mini-PCIe module with an Atheros AR9280 chipset, and the ath9k driver is in use. The WNIC's Transmit (TX) power is set to 17 dBm (50.12 mW) with RTS/CTS and fragmentation threshold disabled. Unfortunately, useful information such as operating frequency, RX sensitivity and default bit rate is not displayed. It will thus be assumed that the actual operating frequency is 5.86 GHz as configured earlier, since there is no way to confirm this without

Figure 4.3: WNIC information and default OCB configuration



performing spectral analysis. Since the WNIC is an OEM module no datasheet for this specific device could be found - datasheets of similar devices such as Compex WLE200NX [95] were consulted to get an approximate specification of the RX sensitivity. Based on this information the RX sensitivity is assumed to be -85 dBm (operation at 5 GHz 802.11n HT20 MCS3 was chosen as a comparison). Lastly, the default bit rate needs to be determined. Unfortunately not even `iwlist` shows the current or available bit rates when the adapter is in OCB mode, as shown in Figure 13. It can however be determined by experimentation.

Summary of the default parameter values are presented in Table 4.2 after the determination of the bitrate.

4.3.1.2 Experimental Set Up and Verification

A basic experimental set up in ideal communication conditions can be used to investigate and characterise direct-link communication, and can be used as a guide for later experimentation. Such a scenario would consist of two nodes communicating directly with each other in LOS conditions over a short distance with no interference from external sources. This basic scenario also doubles as an ideal opportunity to verify implementation and functionality of the tools and utilities, such as Wireshark and `iperf`. Results obtained from a trustful experimental set up can then be used to verify network simulation results by setting up an identical scenario in simulation.

Two nodes are placed 1 m apart on a large table in a closed room with brick walls. No big metal objects were present in the room, no large obstructions were close to the nodes and no other transmissions in the 5 GHz band was detected. Signals will however be reflected off the walls and floor causing a degree of interference, but this effect was assumed to be negligible.

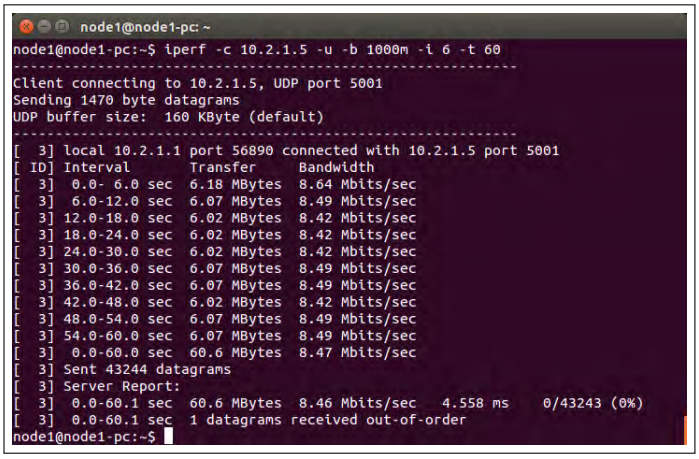
As an initial test, an instance of `iperf` UDP server was run on one node

with the other acting as a client. The client's instance of `iperf` was configured to send UDP data at a rate of 1000 Mbps for a duration of 60 seconds in order to make sure the channel is saturated. Both instances of `iperf` were set to display updates in 6 second intervals, displaying average measured values 10 times per minute. Figure 4.4 displays the commands used to set up the UDP server and client with `iperf` as described. It should be noted that the configuration AODV was *disabled* to eliminate any additional routing overhead (albeit little). `Wireshark` was used to capture and analyse packets during all experiments, with instances on the client and server during this experiment.

Figure 4.4: UDP server and client set up with `iperf`

```
1 Server: $ iperf -s -u -i 6
2 Client: $ iperf -c 10.2.1.5 -u -b 1000m -i 6 -t 60
```

Figure 4.5: 1 Gbps saturation test result with `iperf`

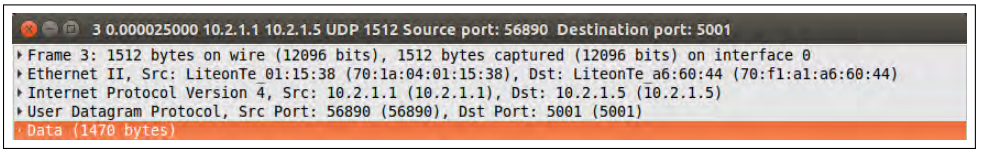


Above mentioned test was repeated five times, with Figure 4.5 displaying the result of one such test on the client side. Across all tests the average throughput measured on server and client side remained constant with a maximum difference of 0.01 Mbps between tests, with the total average calculated to be 8.464 Mbps on the server side and 8.472 Mbps on client side. The server reported an average jitter of 4.517 ms and no lost packets across all tests.

Wireshark was used to validate the throughput results displayed by `iperf`, which resulted in the conclusion that `iperf` displays the average *goodput* (application level throughput). By analysing the UDP conversations with Wireshark it was seen that the total average throughput is instead 8.715 Mbps, with goodput according to this result calculated to be 8.473 Mbps, which confirms the data obtained from `iperf`. The calculation is performed by multiplying

the total throughput by the percentage of useful data per packet. All UDP packets sent by `iperf` have a fixed length of 1512 bytes with a payload of 1470 bytes as seen in Figure 4.6, an extract from Wireshark. The overhead per packet is 2.78% (97.22% useful data per packet), thus yielding the goodput of $8.715 * 0.9722 = 8.473$ Mpbs.

Figure 4.6: `iperf` UDP packet inspected in Wireshark



Unfortunately, all UDP packets sent by `iperf` will have a constant payload size of 1470 bytes irrespective of the targeted throughput. `iperf` will vary the intervals at which packets are sent to achieve the specified average throughput. However, having control over payload size would be required in certain experimental configurations. To achieve this, server and client scripts were written in Python to mimic the functionality of `iperf`, but providing additional features such as changing payload size and automatic graphing of results. These scripts were run in the identical scenario as used above, yielding an average goodput of 8.41 Mbps confirmed by Wireshark.

Now it is confirmed that the implementation and results obtained from `iperf`, Wireshark and the custom Python scripts can be trusted, this set up can be used for future experimentation.

4.3.1.3 Maximum Throughput Comparison

The throughput observed in the previous section, although measured in a saturated channel with ideal communication conditions, is however not necessarily the maximum achievable throughput at the default bitrate. Maximum achievable throughput is not only determined by data bit rate but also by the amount of overhead present when transmitting a packet. Larger payloads will yield better throughput due to constant transmission overhead for all packet sizes.

Evaluating the maximum throughput at various payload sizes and comparing the results to theoretical and simulation data will enable identification of the actual bit rate, as well as verifying the simulation results for this scenario.

The same experimental set up and testing methodology used earlier was utilised to obtain maximum achievable goodput at various payload sizes from 100 B

up to 2300 B in increments of 100 bytes, just shy of the 2304 B Maximum Transmission Unit (MTU) of IEEE 802.11 to avoid fragmentation.

A similar scenario was set up in the simulation environment, however the effect from possible interference from signal reflections was assumed to be negligible (as mentioned earlier) and ignored in the simulation scenario. TX power and RX sensitivity values were updated to match the values determined in the previous section (see Table 4.2 for reference). Bit rates of 9 Mbps and 12 Mbps were evaluated over the same payload sizes as used in the real-world test, with goodput results presented in Figure 4.7.

A combination of approaches used by Wang *et al.* [96] and Song and Choi [97] will be used to calculate the maximum theoretical throughput of IEEE 802.11p at 9 Mbps and 12 Mbps bit rates for payload sizes up to 2300 B. Maximum throughput can be expressed as

$$MT = \frac{8L_{data}}{T_{DATA} + T_{prop} + T_{DIFS} + T_{SIFS} + CW_{avg}} \tag{4.1}$$

assuming an ideal scenario with no interference and errors. T_{DATA} represents the transmission duration for a data packet at a given bitrate, and can be expressed as

$$T_{DATA} = T_{pre} + T_{ph} + T_{sym} * (\frac{16 + 8L_{dh} + 8L_{data} + L_{fcs} + 6}{N_{DBPS}}) \tag{4.2}$$

where N_{DBPS} is the data bits per OFDM symbol for the given bitrate as specified in Table 3.3. Furthermore, CW_{avg} represents the average backoff time, calculated as

$$CW_{avg} = \frac{aCWmin * aSlotTime}{2} (see Table\ 2.2) \tag{4.3}$$

It should be noted that ACK frames are not taken into account when calculating the throughput since ACKs are not in this scenario, which is confirmed with Wireshark. All variable values not explicitly discussed or referenced are presented in Appendix 6.3 Table 2.

Theoretical maximum achievable throughput at bitrates of 9 Mbps and 12 Mbps over payload sizes of 100 B to 2300 B is plotted in Figure 4.7 and compared to real-world measurements and simulation results.

It can be seen that the theoretical maximum throughput yields more idealistic results when compared to the simulation results. Simulation results are marginally lower than theoretical maximum, but do follow the curve of the theoretical results in the case of both 9 Mbps and 12 Mbps.

When comparing the real-world measurements to the theoretical measurements, it can be seen that the real-world results does not align well with either

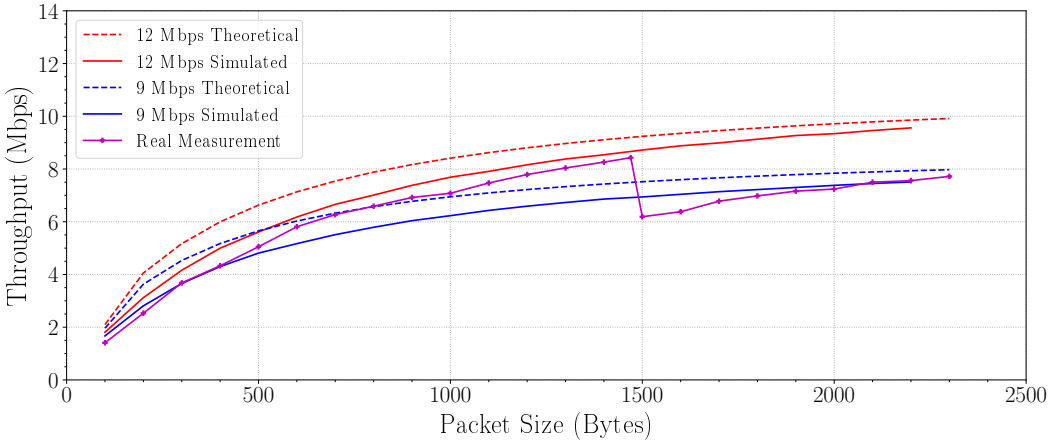


Figure 4.7: Throughput at various packet sizes

9 Mbps or 12 Mbps. However, it is observed that the 12 Mbps simulation results tracks the real-world data well up to the 1500 B mark, although the empirical data's goodput is slightly lower than the simulation. According to these results, a payload length between 600 B and 1500 B is the optimal range in which the simulation provides the most accurate representation of real-world performance, with the maximum difference in throughput results being 5.56%. At 1470 B, the difference was confirmed to be a mere 2.27%.

The sharp drop at 1500 B was caused by the packets undergoing fragmentation due to the 1500 B MTU of Ethernet v2. As seen in Figure 4.6, Ethernet v2 is used when using `iperf` and the custom Python scripts. It is clear why `iperf` specifically uses 1470 B payload sizes for UDP communication, since it is the payload size yielding the best possible throughput when Ethernet v2 is used. Ethernet v2 is not used in the network simulation, and it was confirmed via further testing that the fragmentation threshold in simulation is 2304 B which is the MTU of IEEE 802.11. This threshold is configurable, and can thus be set to mimic the 1500 B threshold observed in the practical implementation.

In conclusion, simulated and empirical throughput data does not match the theoretical calculations. This is to be expected since it is an ideal representation of the maximum performance, and will never be achieved in practice due to various factors such as interference and signal quality. It was observed that the empirical data matches the 12 Mbps bitrate simulation to a fair degree. It can be concluded that the default bitrate of the wireless adapters in this scenario is 12 Mbps, as there is also a close correlation between the simulation and theoretical results. This also provides substantial validation of the PHY and MAC implementation of IEEE 802.11p in the simulation environment. It is confirmed that using a datarate of 12 Mbps to deliver UDP payloads smaller than 1500 B in this simulation set up will yield trustworthy results compare closely to the actual real-world implementation.

4.3.1.4 Contention Comparison

In large scale simulations, nodes will continually contend for medium access which will have a big impact on overall network performance. Investigating the effect of contention in a small-scale ideal scenario, would aid in verifying that this mechanism is correctly simulated.

A third node was added to the same real-world testing scenario used up to this point, with all three nodes being 1 m apart in a triangle formation. Two nodes will act as clients, sending 1470 B data packets to the server node at a targeted data rate of 15 Mbps to saturate the medium. One client will continually transmit data whilst the second client will only transmit data during a specific period, during which both clients will need to contend for medium access.

The simulation scenario was adjusted to mirror the real-world set up. As mentioned earlier, parameters of the NICs in simulation were adjusted to match the default NIC parameters obtained during this section, presented in Table 4.2. Periods during which simultaneous transmissions occurred in the real-world experiment were matched in the simulation for easy comparison.

Figure 4.8 presents the goodput measured at the server node for one real-world experiment (although 5 tests were executed) as well as the same measurement in simulation for a single seed. Results of multiple experiments and differently seeded simulations did differ, but the trends across all tests were the same. Plotting single instances instead of the averaged results, better highlights the characteristics observed during these tests.

Table 4.2: Determined default parameters of WNIC device in OCB mode

Parameter	**Value**
TX power (no antenna)	17 dbm (50.12 mW)
RX sensitivity	-85 dBm
Data bitrate	12 Mbps
Operating frequency	5.86 GHz

In Figure 4.8 it can be seen that simulation and empirical data share similar trends. Goodput is stable whilst a single client is communicating, and starts to fluctuate once the medium becomes contested. Larger fluctuations are observed in the real-world test, but the average goodput over the contested period remains the same as in an uncontested scenario - this was observed during all tests. In simulation the goodput fluctuations share similar characteristics as observed with the empirical data, but the average goodput during the contested period drops by an average of 4% over multiple iterations with different

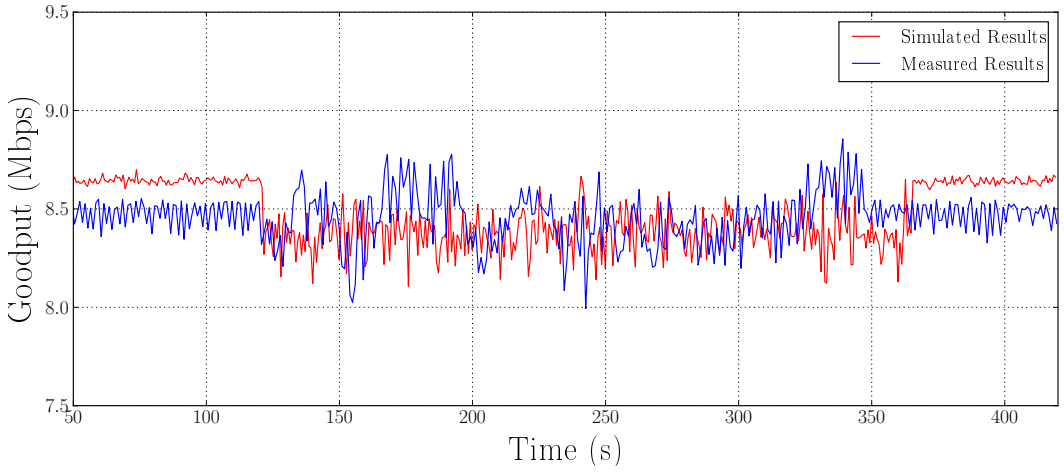


Figure 4.8: Simultaneous communication comparison

seeds.

Due to the trends in goodput fluctuations and average goodput results being similar in a medium contested by two clients, it is concluded that the implementation of contention in simulation is satisfactory and trustworthy.

4.3.2 Goodput over Distance and Signal Fading

All previous experiments were performed in an ideal scenario where nodes were directly next to each other. This scenario is ideal to observe the maximum achievable performance, but such favourable conditions will rarely be encountered in a VANET. Signal fading will negatively impact network performance over distance, even in an ideal LOS scenario as discussed in Chapter 3.2.2. Due to the severe impact this effect has on signal quality and the potential sparsity of nodes in a VANET scenario, it is of high importance that the characteristics of communication over distance is modelled optimally in simulation when an accurate representation of real-world performance is desired. Validation and calibration of the simulation's approximation of communication over distance and maximum communication range is required.

The maximum achievable goodput over distance will be compared between real-world and simulation scenarios to determine the accuracy of the simulation in this regard. Four major factors, namely operating frequency, transmission power, receiver sensitivity and signal fading, will determine the maximum effective communication range and throughput of two static nodes in an ideal LOS scenario (ignoring aspects such as antenna characteristics and interference from external sources). These factors can be individually adjusted such that the simulation environment provides a better approximation of a real-world implementation.

4.3.2.1 Open Field Testing

An ideal location to test the maximum communication range and achievable throughput over distance would be a large isolated grass field. This is ideal since the grassy surface will cause minimal signal reflection and no obstructions or major interference from external sources are present. Such a location, presented in Figure 4.9, was identified and used for testing.

Two nodes were placed in the field at variable distances from each other, on bar stools of approximately 1 m in height. One node was kept at a fixed position, connected to a vehicle parked under trees indicated by the blue circle in Figure 4.9, whilst the second node (connected to a custom battery power source) could be repositioned along the red line. Possible reflections from vehicles parked under the trees were assumed to be negligible.

UDP data packets with a payload of 1470 B were sent at a rate of 15 Mbps from one node to the other, in order to saturate the channel and provide a maximum throughput scenario. Data was sent for 300 seconds in each location to obtain a reliable indication of the achievable goodput. Average results at various distances are plotted in Figure 4.10, compared to results obtained from the same scenario in simulation.

Figure 4.9: Grass field used for range testing. Source: Google Maps

The simulation's approximation of goodput over distance declines gradually over distance with the characteristic of a stretched inverse 'S' (caused by the NakagamiFading model), with 1 Mbps goodput still achievable at 650 m. It is seen that the real-world tests yielded slightly better goodput performance up to 150 m, with goodput remaining close to the maximum achievable value until

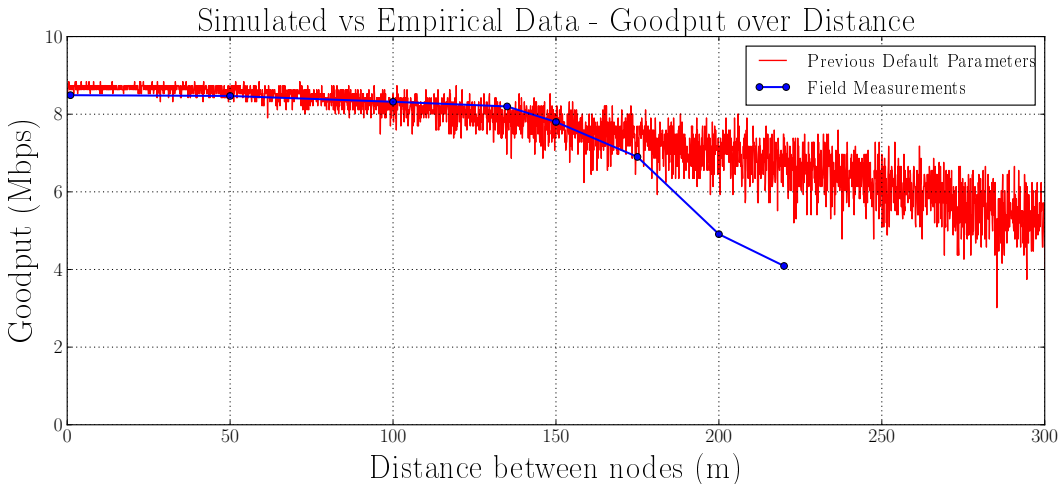


Figure 4.10: Goodput over distance in open field

approximately 130 m. After 130 m it started to decline with a similar inverse 'S' shape, albeit with a more aggressive gradient, than observed in simulation. Overall, the simulation's representation is satisfactory up to the 170 m mark, where after it is too optimistic.

At distances more than 200 m, signal quality and data throughput in the real-world test was severely unstable and unreliable. Large fluctuations were observed in cases where a connection was possible, and in some cases no link could be established when re-running the same test in the same location. At these great distances it was observed that throughput measurements and ability to establish a link became very sensitive to the slightest device rotation or positional movement.

This substantiates why measured results are only plotted up to a certain point in Figure 4.10, with 220 m being the furthest distance at which results were somewhat reliably reproducible in this experimentation scenario, although with deviations from the average of up to 30% in both directions. Figure 4.11 presents extracts from the measured goodput at 135 m and 220 m, highlighting the instability at 220 m compared to 135 m (note the y-axis scales).

As previously stated, slight changes in device positioning greatly influenced performance results at greater distances between nodes. It was concluded that this effect was caused by the antennas' radiation pattern. 'Sweet spots' could be found by tweaking the height and horizontal rotation of the antenna.

To demonstrate the severity of this effect, the antenna of the node in the field was subjected to 360 degree rotation on its horizontal axis, starting in an upright position perpendicular to the horizontal plane. A separation distance

Figure 4.11: Measurement extracts at 135 m and 220 m

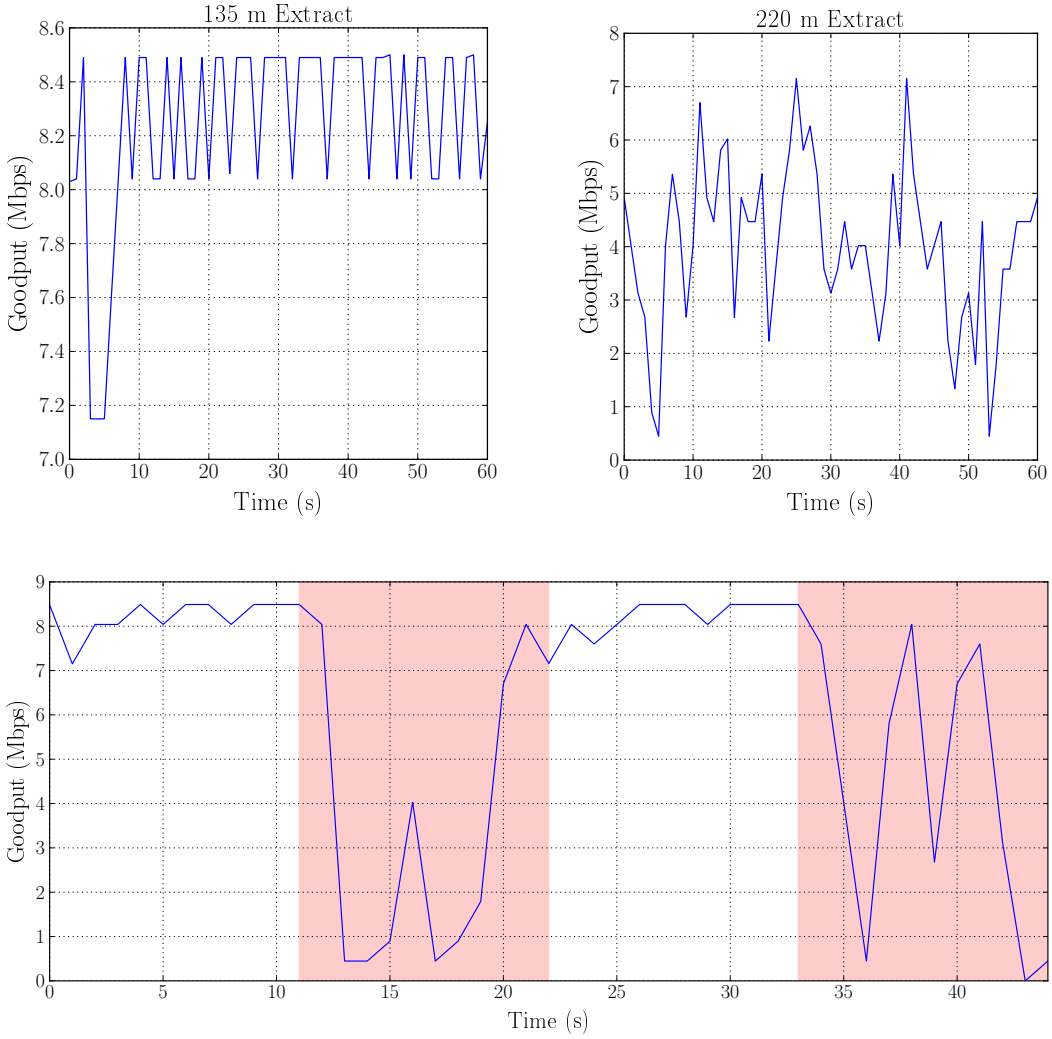


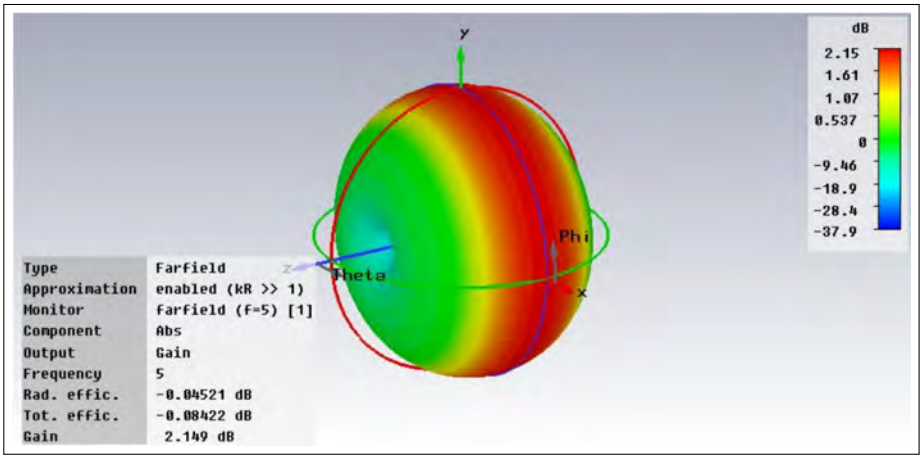
Figure 4.12: Horizontal rotation of antenna at 120 m

of around 120 m between nodes yielded the best visualisation of this effect, presented in Figure 4.12. Areas highlighted in red indicates the periods during which the rotating antenna was orientated perpendicularly to the stationary antenna (thus in the areas around 90 and 270 degree rotation), resulting in major performance losses.

At larger distances this effect is magnified to the extent where the windows for successful communication is only a few degrees wide. A similar effect was observed when adjusting the height of the nodes, with the vertical 'band' in which optimal performance can be achieved getting smaller as the distance between nodes increase.

This phenomenon can be explained by investigating the antenna radiation pat-

Figure 4.13: Radiation pattern from Tareq *et al.* [5]



tern. Unfortunately, no documentation for the specific antenna model could be obtained, since the only information received from the supplier was specification for operating frequency and gain. Since this antenna type is visually similar to those used on 2.4 GHz WiFi routers, it is assumed that the antenna is a dipole.

A simple dipole antenna consists of two in-line conductors of identical length, connected to a central feedline. A 'half-wave' dipole, the most common dipole type, has a total conductor length equal to approximately half of the electrical wavelength it is designed for.

Tareq *et al.* [5] presented a half-wave dipole antenna design with a 5 GHz resonant frequency. Therefore it is assumed that the radiation pattern of this project's antennas will be similar to the antenna in [5]. Figure 4.13, an extract from [5], shows the far-field radiation pattern, with the antenna positioned along the z-axis. Red areas indicate maximum gain (in dB), whilst green areas indicate lower gain.

One of this project's antennas was disassembled and measured, as depicted in Figure 4.14. By visual inspection, the antenna is confirmed to be a dipole. It can be seen that the measured length of the conductors is 29 mm and 23 mm respectively, with a total length of 52 mm. Therefore it is concluded that the antenna is indeed a dipole, but not exactly a half-wave dipole.

Aristotle Enterprises Inc manufactures an antenna, RFA-02-L2H1, that has similar physical properties to the antennas used in this project, and a gain of 2 dBi. This antenna is designed for 2.4 GHz, but the radiation pattern will not differ a lot at 5 GHz. Figure 4.15, an extract from the datasheet of RFA-02-L2H1 [6], shows the antenna's radiation patterns. It can be seen, especially when looking at the vertical polarisation pattern, that there is a lot

Figure 4.14: Disassembled antenna

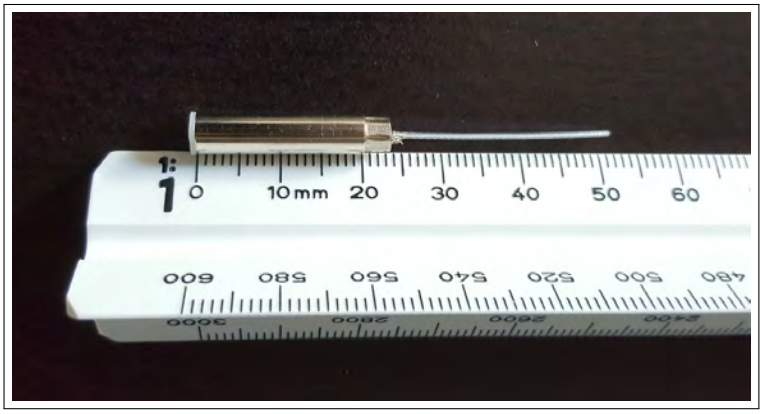
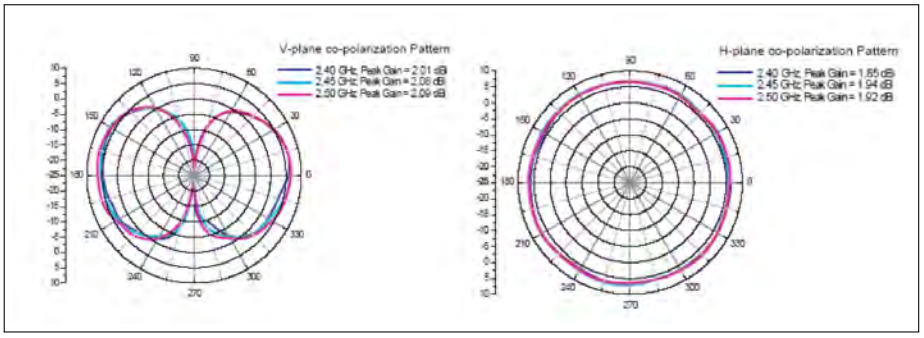


Figure 4.15: RFA-02-L2H1 radiation patterns [6]



of similarity to Figure 4.13. It is therefore assumed that the antennas used in this project have similar radiation patterns.

Based on this assumption and the experimental results discussed earlier, it is concluded that the characteristics of radiation pattern shown in Figure 4.13 correlates with that of the measurements displayed in Figure 4.12. Therefore it is assumed that the radiation pattern of the antennas is responsible for the trends depicted in Figure 4.12.

These experimental results gave insight into the importance of appropriate antenna selection for a given scenario. It is concluded antenna characteristics play a critical role for achieving reliable performance in a VANET scenario, and should be regarded more highly in the future.

Since OBUs will never be mounted at identical heights and orientation due to vehicle differences, and because the height differences could be induced by terrain topology, selecting an antenna solution suitable for these scenarios is critical success factor for OBU design.

4.3.2.2 Adjusting Simulation Parameters

Simulation approximation of communication over distance is too optimistic after the 180 m mark when compared to empirical data. A semi-pessimistic approximation in simulation would be more favourable than an optimistic one, since overestimating the maximum network performance will provide a misleading picture of what could be achieved when actually deploying the solution. Adjustment of current simulation parameters is required to obtain a more realistic (or pessimistic) approximation.

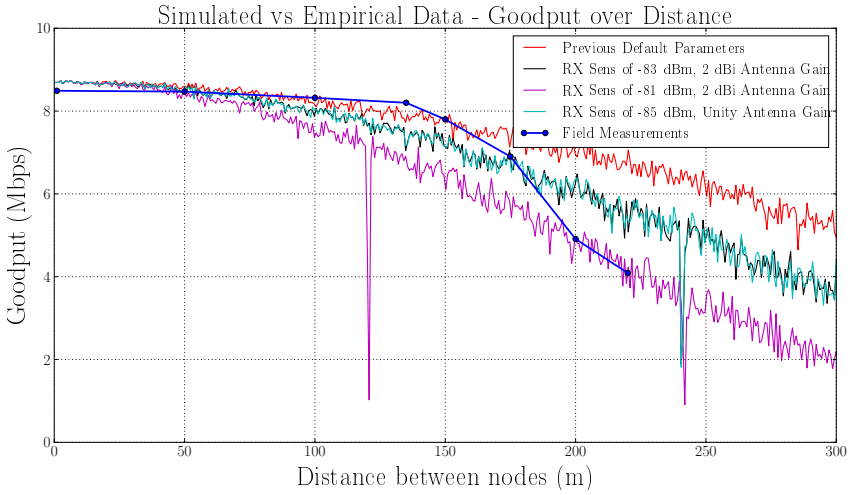


Figure 4.16: Adjusting RX sensitivity

As a refresher, TX power and RX sensitivity parameters shown in Table 4.2 were used in the simulation set up, with an isotropic antenna providing 2 dBi gain in all directions. It would be ideal to have a more accurate antenna model, but no suitable option is available in INET at this time. Using this ideal approximation of an antenna is satisfactory in this project since all nodes are assumed to be at identical heights in simulation and terrain topology is not modelled. These assumptions and idealities do however need to be kept in mind when analysing and comparing simulation results.

RX sensitivity of -85 dBm is the only assumed parameter as discussed in Section 4.3.1.1, so adjustment of this parameter is the logical first step. Adjusting the antenna gain or RX sensitivity (equal but, inverse effects as seen in Figure 4.16) will be immaterial overall. Adjustments were made and simulation results are presented in along with the original data from Figure 4.10. It should be noted that a windowed average was applied to the data to ease the comparison.

As seen in Figure 4.16, changing the RX sensitivity to -81 dBm yields a better approximation of the achievable performance at longer ranges. It does however underestimate the goodput from 50 m to 180 m, but it was argued that it is more favourable to yield a pessimistic approximation at closer ranges whilst providing a more realistic approximation at further ranges. It was argued to be favourable since nodes in a real VANET deployment will mostly be sparsely populated with large distances between them. It is also more optimal to present worst case results rather than overly optimistic results when analysing the feasibility of a solution.

A counter argument would be that no adjustment is needed in this case, since the area of interest is urban, and LOS ranges are short. Nodes will anyway not be able to communicate at longer ranges due to LOS being restricted by large buildings, focusing the approximation parameter choice on short distance performance.

It was chosen to proceed with the first argument (sensitivity adjustment), and choosing -81 dBm as the new RX sensitivity value with TX power and antenna gain remaining unchanged.

4.3.2.3 Street Scenario Testing

VANETs are of course not deployed in fields, but on roads. Even when direct LOS is possible between two nodes, signal reflection caused by the road, buildings and adjacent vehicles will have an impact on performance. A street scenario, depicted in Figure 4.17 was used to observe the extent of such effects and compare the results to those measured in the open field scenario.

This specific street was chosen for two major reasons: 1) During the day many vehicles are parked on both sides of the road, and with it being a narrow one way street, a perfect 'worst case' LOS scenario is created as seen in Figure 4.18. This 'through way' between rows of vehicles will create ample opportunity for signal reflection. 2) Very little traffic passes through this street, which means measurements can be easily collected and repeated without interference from oncoming vehicles.

The experimental scenario is as follows: A stationary node is placed on a high chair as seen in Figure 4.18, whilst a secondary node will traverse the street at a speed of 1 m/s (this speed was chosen for convenience of travel distance calculation) maintaining LOS with the stationary node. UDP data packets with 1470 B payload are sent to the stationary node at a rate of 15 Mbps and the goodput is measured. This experiment was performed several times with the total distance between nodes being up to 140 m. The main thread during all the tests was the stability of throughput at the maximum achievable value, with some reproducible variations and dips during specific

periods. The measured throughput matched that of Figure 4.10.

Figure 4.17: Street scenario

Figure 4.18: Street with cars and hardware setup

Figure 4.19 presents the goodput over time for one test of interest (keep in mind that in this case the time also represents distance between nodes). Areas highlighted in red are the reproducible variations mentioned earlier, with the same characteristic observed during each test. These effects were observed at locations along the street where vehicles were parked on both sides, and is assumed to be the result of signals reflecting off the metallic vehicle bodies.

Based on this observation it can be concluded that such effects would have a notable impact on performance in a VANET, especially in central Stellenbosch where vehicles are parked in almost every street.

This specific test was chosen as an example, since during the test a pickup truck drove down the street and blocked LOS for the period indicated by the blue area in the figure, resulting in a complete loss of signal. Up to this point it was not realised that a relatively small obstacle such as a vehicle will have such a severe impact on the signal.

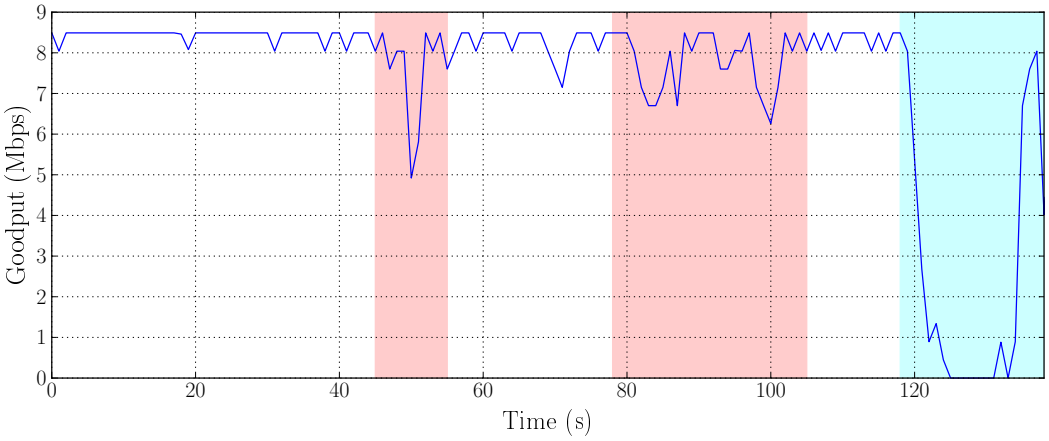


Figure 4.19: Goodput of mobile node whilst traversing the street

Observations made during this experimentation highlighted the major impact vehicles could have on network performance. And since this is a VANET application, such scenarios will always be encountered. To account for such effects in simulation, nodes in the simulation have to be represented as moving obstacles which is unfortunately not possible with INET due to the manner in which obstacles are implemented. Parked vehicles can however be modelled, but this would have to be done in a semi-manual fashion and time limitations restricted this implementation.

Although having a great effect on network performance, obstacle models of stationary and moving vehicles will be omitted from the simulation set up. This should however be kept in mind when assessing and analysing network performance, especially in dense traffic scenarios.

4.3.3 Impact of Obstacles

Lastly, the implementation of obstacles (aka buildings) in simulation will be compared to empirical measurements. It was assumed that all buildings will be modelled as solid polygons with homogeneous material characteristics and a height of 10 m. This is a very simple approximation since all buildings have variable characteristics such as wall thickness, material properties, height and contents of the building itself.

Detailed modelling of obstacles is not within the scope of this project, but attempts were made to calibrate the above mentioned implementation such that it at least provides a decent approximation of empirical data in certain scenarios.

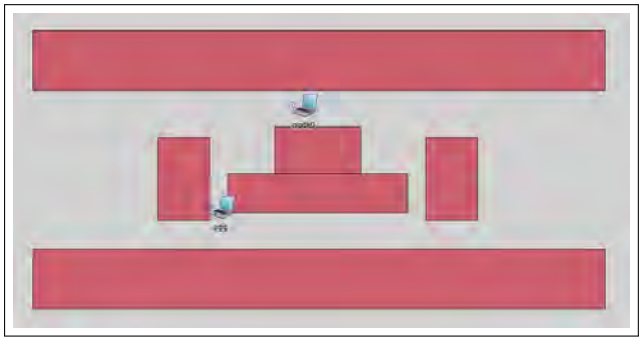
An infinite amount of possible scenarios can be investigated in the central Stellenbosch area and it is not feasible or practical to cater for all possibilities. The chosen area in which to investigate the effect of obstacles is of simple nature

in order to easily recreate it in simulation. An aerial overview of the area of interest is provided in Figure 4.20.

The experimental scenario is as follows: A stationary node is placed at the blue dot, indicated on Figure 4.20, whilst a secondary node will travel at a constant speed along the route indicated by the green line. Once it reaches the red dot it will remain stationary for 10 seconds, and ten proceed to move back to the origin along the same route and speed.

Figure 4.20: Real-world obstacle scenario - blue dot represents the stationary node, green line is the mobile node's path of travel, red dot is the intermediate resting location of the mobile node

Figure 4.21: Simulated obstacle scenario



UDP data packets with a payload of 1470 B will be sent to the stationary node at a rate of 15 Mbps, and goodput will be measured. This scenario was

approximated in simulation as shown in Figure 4.21. Various material properties were applied to the obstacles and compared to the empirical data. Five iterations of the test were performed, and results are presented in Figure 4.22.

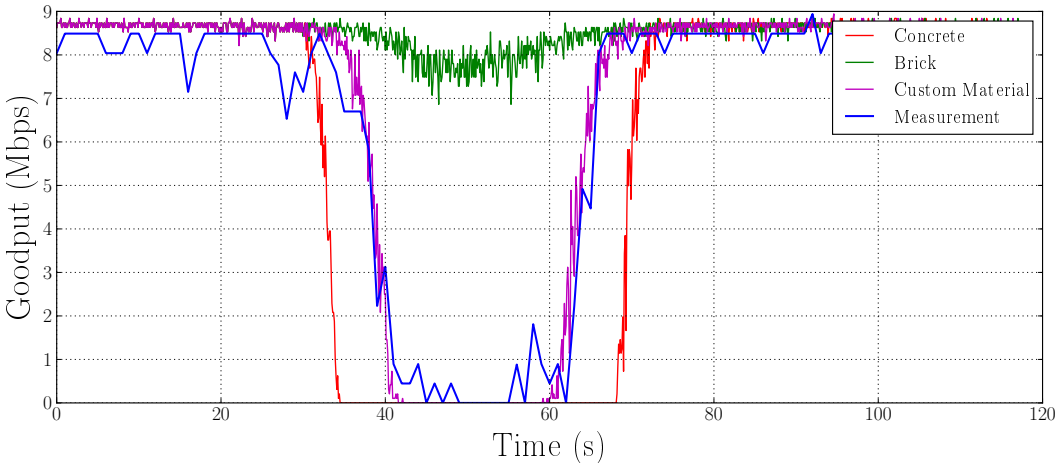


Figure 4.22: Obstacle material comparison

Brick and concrete materials are pre-made material properties available in INET, as discussed earlier in this document, and it can be seen that concrete provides a much better approximation than the brick material with brick having a very small effect on the signal strength. Between 35 and 70 seconds LOS was completely broken, with the node being stationary at the red dot from around 45 to 55 seconds. In the case of concrete it is apparent that the throughput drops at an extremely sharp gradient once LOS is broken, which is not the case with the empirical measurements.

A custom material was created to improve upon the aggressive result yielded by the concrete material. This material's resistivity, relative permittivity and relative permeability values are 550 Ohm, 4.5 and 1 respectively. It provides a better approximation of the given scenario, as seen in Figure 4.22, although it is still not perfect.

During the stationary period, spikes in throughput is observed in the real-world test, which is assumed to be the result of signal reflections from the adjacent wall. This effect does not seem to be accounted for by the obstacle model as seen in the case of concrete and the custom material.

In conclusion, concrete material will be an ideal choice if a pessimistic approximation is required, whilst the custom material provides a more accurate approximation based on this specific scenario. The custom material will thus be used in all further simulations unless otherwise specified.

4.4 Summary

In this chapter, a hardware solution fully implementing IEEE 802.11p (PHY and MAC) was successfully created by using a single board computer, wireless mini-PCIe adapter with an Atheros A9280 chipset and a custom Linux kernel implementing an adjusted MAC layer and device drivers. AODV-UU was modified to operate on the same kernel version, resulting in a 802.11p-enabled hardware solution capable of multi-hop data transfer. This hardware solution was used to successfully verify and calibrate core aspects of the network simulation environment by means of experimentation in the field.

This solution can be used by other researchers in the field in need of an affordable, open-source implementation of a VANET OBU. Source code of the modified AODV-UU implementation was made available publicly for any researcher that is in need of implementing AODV-UU on a Linux kernel version newer than 2.6.

Chapter 5

Simulation of Distributed Licence Plate Detection Implementation

Distributed licence plate detection implemented in a VANET was identified as a possible aid for the combating vehicle-related crimes in South Africa.

Investigating the suitability of IEEE 802.11p and AODV for deployment of this solution in an urban VANET scenario will be performed with simulation. It is desirable to observe the operation of the implementation on a large scale, and, as mentioned in Chapter 2, commercial or custom hardware deployment is not a feasible option in this project.

A simulation environment implementing the necessary networking standards and protocols was set up in Chapter 3 in conjunction with an approximated implementation of real-world vehicle mobility patterns and physical obstructions in an urban area. Aspects of the network simulation model, such as achievable throughput over distance, were verified and calibrated in Chapter 4 by means of experimentation with a custom-made hardware solution.

Implementation and analysis of distributed licence plate detection and reporting in the calibrated simulation environment for a given scenario will be discussed in this chapter.

5.1 Creating an Implementation and Scenario

Distributed licence plate detection and reporting realised in a VANET is envisaged to operate as follows: Vehicles are equipped with licence plate detection hardware and ad hoc wireless communication hardware implementing IEEE 802.11p and AODV. As the vehicles drive around, they will extract licence plate information from surrounding vehicles and log relevant data from each detection (licence plate information, estimated location of detected vehicle de-

rived from an on-board GPS unit as well as the time at which the detection occurred).

Vehicles equipped with this hardware, called active nodes, will attempt to periodically deliver the logged data to a central point by means of multi-hop data transfer in the wireless ad hoc network. In this project, the central point will be a RSU (called an aggregator) connected to back-end infrastructure responsible for processing the aggregated data. Authorities can then extract relevant information from this data for use in criminal investigations or response unit deployment to intercept a wanted vehicle detected by the system. Such a system will be standalone and functionally independent from other fixed infrastructure and ITS systems.

As previously mentioned, focus will be placed on the communication network aspect of this system - implying that the area of interest is data communication within the VANET itself. The investigation of accurate licence plate detection and processing of aggregated data in back-end infrastructure does not fall within the scope of this investigation.

5.1.1 Application Design

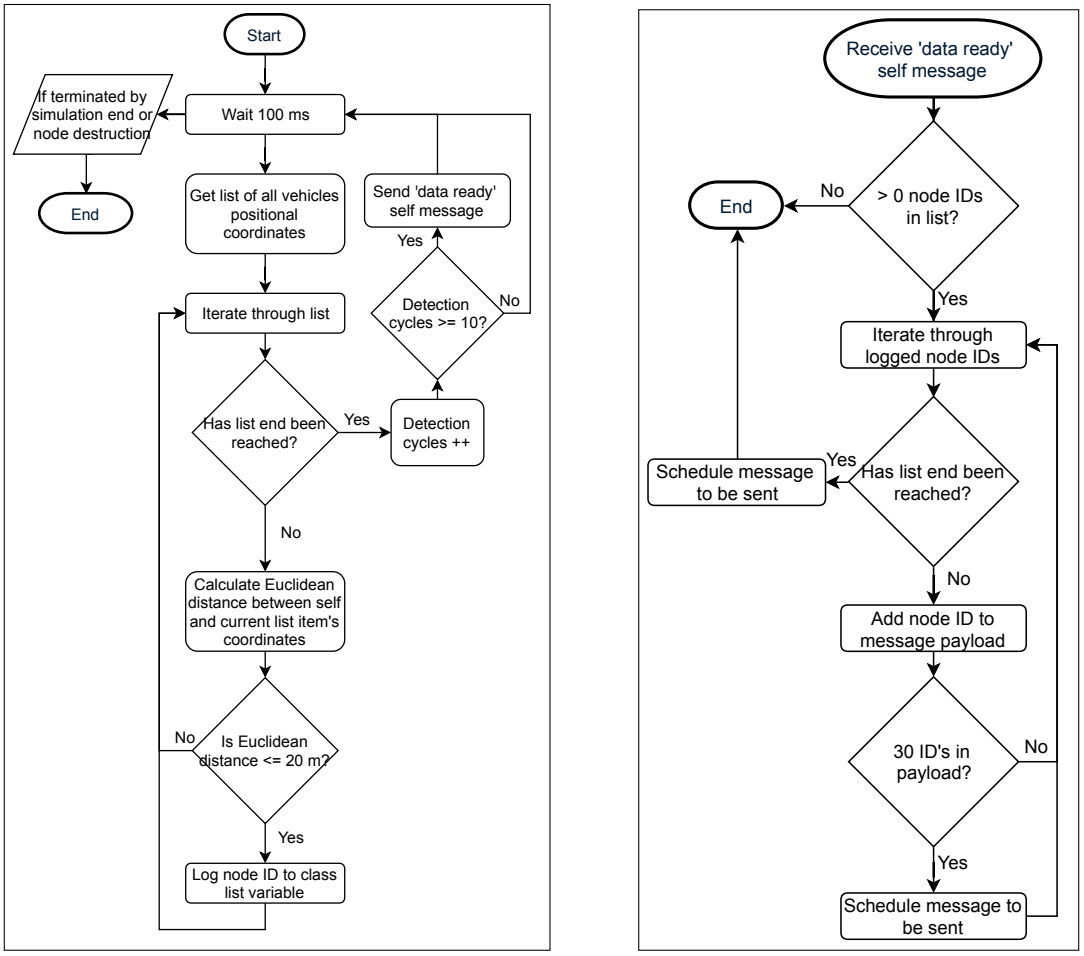
A simulation application needs to be created to appropriately generate data to fit this scenario since data generation is a unique characteristic of the envisaged system. The quantity as well as the frequency of generated data is dependent on the detection of vehicles, thus the characteristics of traffic flow within the scenario will have a direct influence on network traffic.

Having random nodes sending random data at random intervals is a valid approach to benchmark the maximum achievable performance of a VANET, but the objective is to specifically investigate the operation of the abovementioned implementation.

This application will consist of two components, namely vehicle spotting and data traffic generation, and functions as follows:

Vehicle detection. Each active node scans the simulation environment 10 times per second and logs the licence plate information (in this case the node ID) of each node within an Euclidean distance of 20 m from its current location. A distance of 20 m was chosen based on parameters used in [98], but the effective distance of an actual LPR implementation is dependent on the hardware in use as well as scenario specific conditions such as weather and lighting. Figure 5.1 (a) provides a flow diagram detailing the operation of the vehicle detection algorithm.

Figure 5.1: Application design



(a) Vehicle detection algorithm (b) Data generation algorithm

This detection algorithm is not an accurate reflection of an actual LPR system, since vehicles can be detected in all directions with no regard to the orientation or speed of target vehicles. LOS is also disregarded, meaning vehicles behind buildings and all vehicles in traffic will be spotted. These assumptions will be taken into account when drawing conclusions from simulation data.

Data generation. After 10 'LPR' scanning cycles the node will transmit the logged data with a burst of UDP packets containing 1470 B payload each. UDP was chosen due to its small overhead which would be favoured in a scenario with a large number of nodes, as well as the fact that the data is not critical or sequence dependent.

In simulation, payloads will be filled with random data but the quantity of packets sent during the burst will be determined by the amount of detections. A static payload size of 1470 B was chosen to operate at the point of maximum achievable throughput, as discussed in the previous chapter. Choosing a

larger packet size will also reduce the number of transmissions needed to send data, possibly reducing the number of collisions and medium contention in the network.

We assume each detection generates data of 50 byte length: Licence plate information of 10 character length (10 bytes), latitude and longitude of the detection (24 bytes), Unix timestamp (10 bytes) and 6 bytes allocated to delimiters. Data of 29 detections can thus be fit into a 1470 B payload, therefore, if 15 vehicles are detected a single 1470 B packet will be sent and if 30 vehicles are detected two 1470 B byte packets will be sent.

The amount of data sent will thus not be a direct reflection of the amount of data generated, but, as previously mentioned, all packets were kept at a constant size. Figure 5.1 (b) provides a flow diagram detailing the operation of the data generation algorithm.

5.1.2 Simulation Scenario

In-depth analysis of any VANET implementation is a multi-dimensional problem. Copious amounts of variables such as node density, traffic flow, geographical location, aggregator placement, etc. will impact results generated during simulation. It is not feasible to investigate every possible option and variation of such parameters in this project due to time constraints, thus a specific scenario set up will be used to investigate the problem at hand.

Major trends, characteristics and traits identified when evaluating the implementation in the given scenario will however hold true for deployments in other similar scenarios, and conclusions will be made as such.

Simulation environment. The calibrated network and traffic simulation environment created during Chapter 3 and Chapter 4 will be used to implement and investigate the application. Specific parameters used for network simulation are presented in Appendix 6.3 Table 3.

Geographical location. An urban area, specifically the urban core of central Stellenbosch, was chosen as the geographical area in which the implementation will be evaluated as discussed in Chapter 3.3. The SUMO road network maps as created in that chapter will be used for traffic simulation (see Appendix 6.3 for more detail).

Traffic generation. SUMO trips were generated for the road network map as described in Appendix 6.3. A total of 3000 vehicles is set to depart between 0 and 600 seconds at a constant rate of 5 vehicles per second (thus a new vehicle will enter the simulation every 200 ms). This means that the network

simulation can include up to 600 seconds of traffic flow, starting at 0 seconds.

Obstacles. Physical obstructions were implemented as per discussion in Section 3.3.3 with material properties as explained in Section 4.3.3. It will be explicitly stated if obstacles are not in use in a specific experiment.

Aggregator placement. For this scenario, a singe point of data collection in the form of an RSU data aggregator was chosen. This aggregator will have the same network model as all other nodes in the simulation, except that it has a static mobility model and is also a data sink (it implements the AODV routing protocol but does not forward packets). Placement of the aggregator will determine results obtained from the network simulation, since it is the destination for all data packets in the network. Although a certain optimal point of placement can be found by analysis and multiple iterations of the scenario, it is not the goal of this project to obtain the best possible location for aggregator placement. Instead, a suitable location was chosen by means of SUMO simulation inspection and knowledge of inherent Stellenbosch traffic patterns. The intersection of Merriman Avenue and Bird Street was chosen, as indicated in Appendix 6.3 Figure 15. It is known that most traffic passing through Stellenbosch crosses this point, and this held true for the SUMO simulation as well. Figure 14 indicates the aggregator placement as in the final simulation scenario.

Simulation time. Simulation time limit is set to a default of 120 seconds (thus the first 120 seconds of the traffic simulation will be used), unless stated otherwise for specific experiments. This limit was chosen by means of inspection as it provides a suitable point between real-world execution times, and observed 'steady state' characteristics due to the trends of traffic flow generated by SUMO.

Application. The application discussed in the previous section was used during all experiments, unless stated otherwise.

5.2 Implementation Analysis

In this section the discussed scenario is investigated and evaluated in terms of various performance metrics such as packet loss rate, unique detection percentage, throughput, etc., to determine suitability of the scenario for implementation of the envisaged solution.

Table 5.1: Penetration rate vs total active nodes

Total Active Nodes in 120s Simulation							
Penetration rate (%)	5	10	15	20	25	30	35
No. of active nodes	32	65	94	116	147	180	209

5.2.1 Analysis

Simulations of this scenario are performed at various penetration rates, meaning that as each vehicle is initialised in simulation it has a probability of detecting as well as forwarding data to other nodes. In a real-world deployment of the envisaged system, not all vehicles will be active nodes. The penetration rate feature was therefore implemented to yield a more realistic representation. This is important since the ratio of active to non-active nodes in traffic will determine the data generation patterns in this specific application, as mentioned earlier.

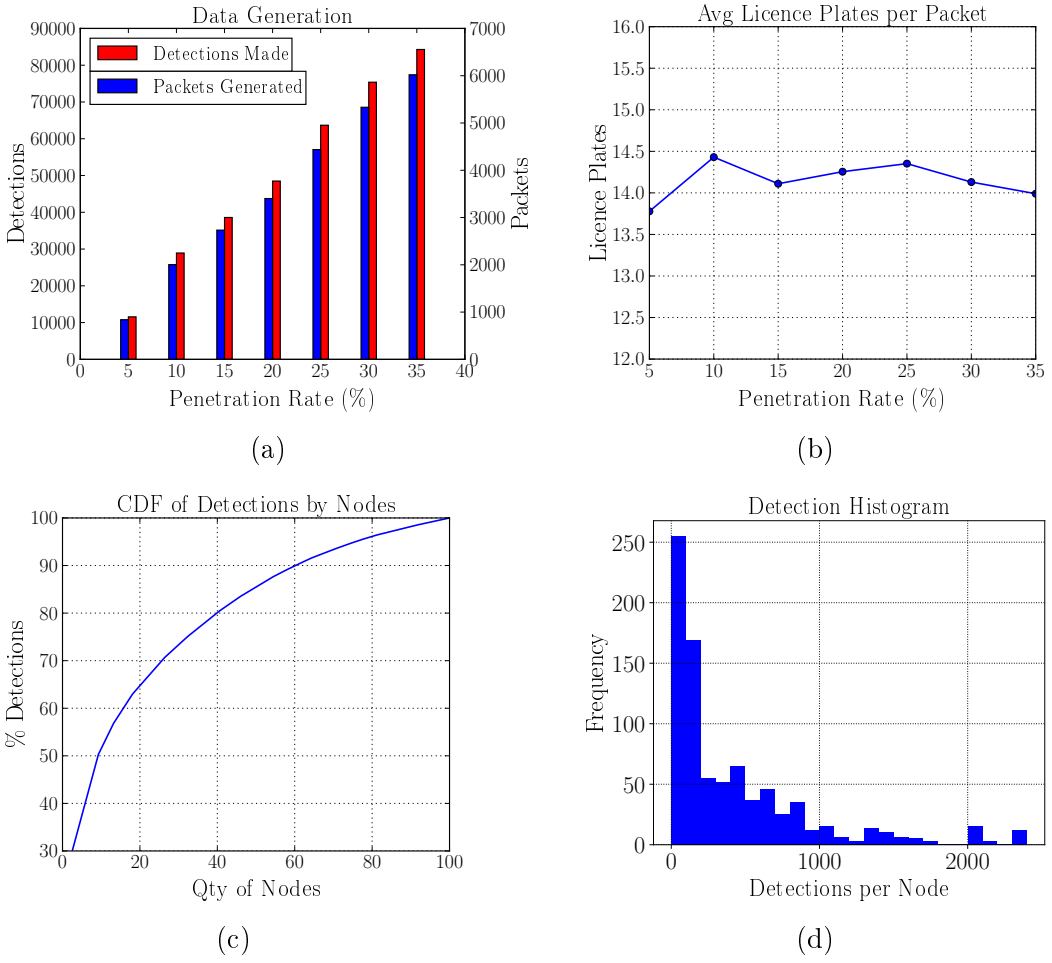
Table 5.1 indicates the number of active nodes present in the simulation for the full duration of 120 simulation seconds. As mentioned earlier, a new vehicle enters the simulation every 200 ms meaning that during the 120 second simulation a total of 600 vehicles will be added. It has to be noted that vehicles entering close to the 120 second mark will not spend much time participating in the simulation (this has to be kept in mind when analysing the data). A penetration range of 5% to 35% was chosen to be representative of possible large-scale deployment scenarios.

5.2.1.1 Data Generation

Figure 5.2 (a) shows the total number of detections made by active nodes as well as the total number of data packets generated in the entire network during the 120 second simulation time for various penetration rates. *Detection* is defined as the successful detection and logging of a vehicle by an active node. A linear relationship can be seen between the penetration rate and the number of detections, with an increase in active nodes resulting in an increase of detections to the same degree. This in turn invokes a similar linear increase in data traffic as expected, since data generation is dependent on the number of detections.

It can be stated that each active node added to the scenario will, on average, generate the same amount of data over time irrespective of its positioning and route. However, more detections do not necessarily correlate with the amount of unique data being generated. Figure 5.2 (d) depicts a histogram of detections per active node across all penetration rates, with Figure 5.2 (c) presenting the corresponding cumulative distribution function. By inspecting these figures it can be seen that a small number of nodes are responsible for

Figure 5.2: Data generation



a large portion of detections. The majority of nodes, however, provide little detections in comparison, with the average detections per node being around 400 across all penetration rates.

The reason why 20% of the nodes generate 60% of the data comes down to traffic characteristics. Some vehicles will sit in traffic and detect the same vehicles repeatedly whilst other vehicles will traverse unpopulated areas, resulting in more unique, but over all less, vehicle detections. This will be discussed in more detail in an upcoming section.

It is interesting to note that even in the case of a large scale deployment of 209 active nodes in the relatively small area of 2×2 km, the useful data generated during 120 seconds is merely 4.21 MB (assuming 50 B data per detection for 84 279 detections). Assuming 24 hour operation at this rate, 3.03 GB of data is generated per day (not a likely scenario since traffic characteristics vary greatly during this period, but it is a good indicator). This amount drops to 416.16

MB for a smaller deployment of 32 vehicles. Relatively speaking this is not a large amount of data for the amount of useful information that can possibly be obtained, making this application scenario promising from a deployment point of view. Keep in mind that traffic patterns and the detection algorithm is not an accurate representation of a real-world system, but a sufficient starting point from which to test the networking side of the problem.

However, the total size of the actual generated data packets is 8.85 MB (6020×1470 B) for the case of 209 active nodes, more than double the expected value. Figure 5.2 (b) showcases the effect by depicting the average amount of detection information sent per packet. Instead of achieving the maximum 29 detections per packet, the average is consistently 14 to 14.5 detections per packet across all penetration rates. This is due to the ways in which packets are generated and sent. These methods were selected to ensure a high throughput as well as to limit the number of transmissions needed for sending the logged data.

It would thus be possible to double the amount of useful data sent with the same network utilisation if the packet generation algorithm is optimised.

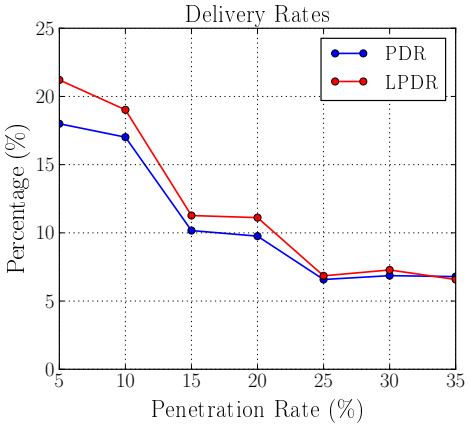
5.2.1.2 Data Communication

Figure 5.3 (a) shows the Packet Delivery Rate (PDR) and Licence Plate Delivery Ratio (LPDR) over penetration rate for the given scenario. In this context, PDR is defined to be the ratio between data packets received by the aggregator without error and total packets generated in the network. Subsequently, LPDR is defined to be the ratio of licence plates (or detection data points, i.e. data logged when spotting a vehicle) successfully received by the aggregrator to the total number of detections made during the simulation.

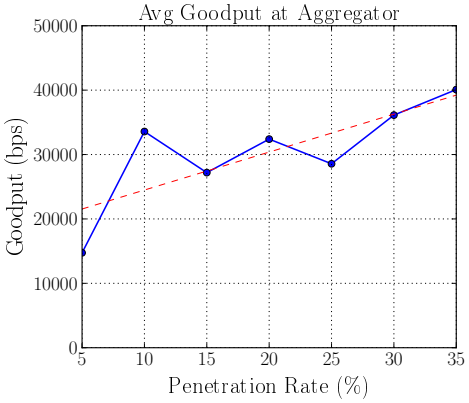
The PDR and LPDR declines as the amount of active nodes in the simulation increases, until it reaches a steady state of about 6.8% at 25% penetration rate. Both PDR and LPDR remain steady despite the increase in active nodes and generated data. The reason for this is an equilibrium is reached between data generation and network connectivity at a certain point.

LPDR is better than PDR at lower penetration rates, meaning that received packets contain more licence plate data on average than at higher penetration rates. This can be attributed to traffic patterns, aggregator placement and active node distribution in this scenario. Since the aggregator is placed in an area where a lot of traffic passes through and active nodes are sparse, nodes that have a route to the aggregator will be in its general geographical area, and will naturally yield more detections per packet due to the higher traffic concentration. If active nodes are in an area with sparse traffic flow, chances are that in the case of low penetration rate these outlier nodes will not have a route to the aggregator, resulting in their packets being lost.

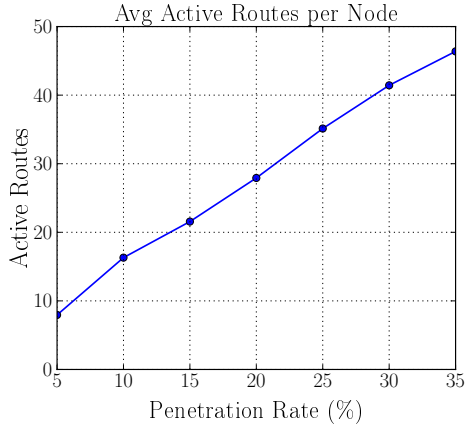
Figure 5.3: Data at aggregator



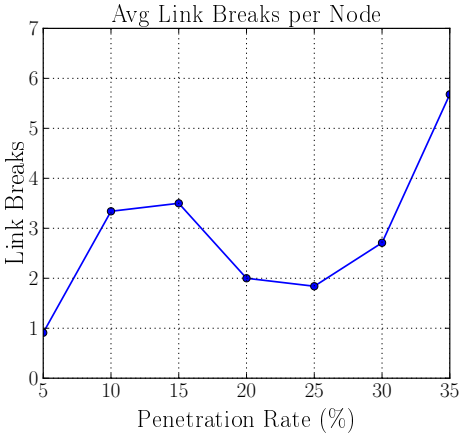
(a)



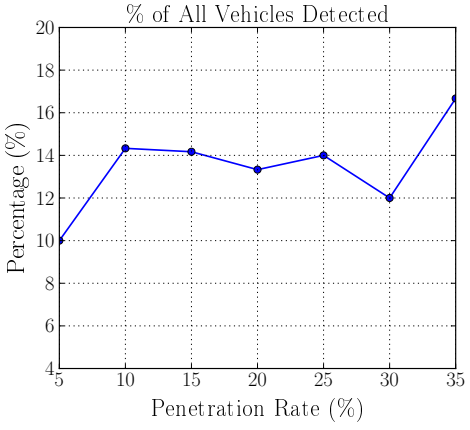
(b)



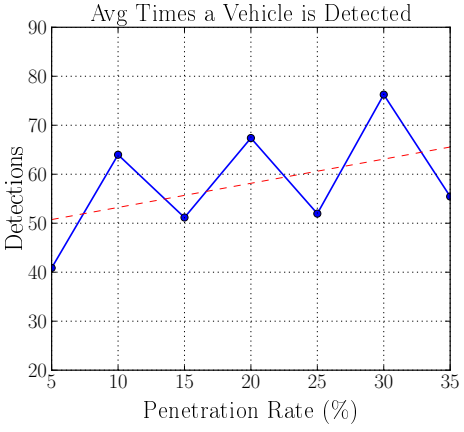
(c)



(d)



(e)



(f)

At its best, the PDR is a mere 18% in the case of 5% penetration rate. More than 80% of packets are lost, which seems odd for a scenario where there would not be many collisions or contention. On the contrary, in scenarios with more active nodes, such factors may cause the low PDR.

Even though the PDR declines to low values at higher penetration rates, the quantity of packets received by the aggregator increases linearly. The number of times the same licence plate is received at the aggregator increases at a linear trend due to this factor, as seen in Figure 5.3 (f). However, the chance that a vehicle is detected at least once during the entire simulation remains relatively constant across all penetration rates, as seen in Figure 5.3 (e).

Due to the fact that the percentage of vehicles detected at least once remains constant and the frequency of detecting the same vehicle is increased, it is deduced that data diversity per spotted vehicle is increased.

In conclusion, an increase in active node penetration rate results in decreased overall network performance in terms of PDR, but yields more diverse data per spotted vehicle whilst the chance of a vehicle being spotted remains the same. Based on these results it is argued that the optimal point of cost vs performance would be reached at a penetration rate of 10%. At higher penetration rates, the only benefit gained would be the frequency and diversity of per-vehicle data given that the vehicle was spotted. In this scenario, it would be favourable to have a large number of vehicles detected a few times rather than having a small number of vehicles detected many times. This improves the chance that a wanted vehicle or vehicle of interest is logged at least once.

If the PDR can be increased, it will result in a potential increase in both spotted vehicle diversity and frequency of spotting data per vehicle. The cause(s) of low PDR in this scenarios needs to be determined.

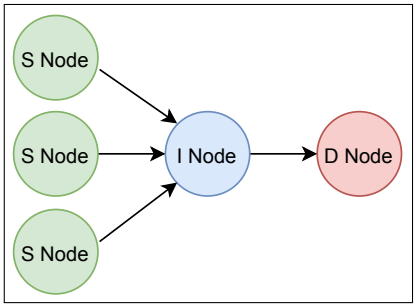
In this application, causes resulting in failure of data to reach the aggregator are as follows: 1) No route possible due to physical isolation or obstacles; 2) link saturation (at aggregator or intermediate node); 3) link breakage due to vehicle mobility or obstacles; and 4) no opportunity to access channel due to contention or bit errors caused by collisions or interference.

Link saturation and congestion. Saturation occurs when a link's maximum communication capacity is exceeded. This happens when the number of packets sent or received exceeds the capacity supported by the link. Packets will be placed in queues to be processed when possible, and if the queues fill up packets will be dropped. Link saturation or congestion can thus cause packet loss, increased latency and decreased data throughput due to additional overhead. These effects can, however, vary in different scenarios.

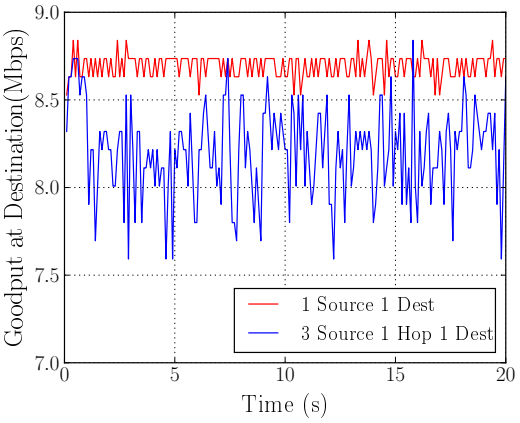
In the 'ideal' scenario, where a link between two nodes is saturated by direct

one-way communication (data solely sent by one node and received by the other). However, it is reasoned that this type of ideal one-to-one communication would be uncommon in this project's scenario, especially where nodes cluster together. A more likely scenario is that multiple nodes send data via a single node, while this node is relaying the data to another destination and has to send its own data too. This is depicted in Figure 5.4 (a), where 'S Nodes' are source nodes, 'I Node' indicates the intermediate node (or hop) and 'D Node' indicates the destination. Data from all the source nodes will reach the destination node via the intermediate node. In this case the source nodes are only transmitting, the intermediate node is transmitting and receiving (possibly within the same time frame) and the destination node is only receiving.

Figure 5.4: Critical link saturation scenario



(a)



(b)

If a node has to send and receive data, e.g. if the node is an intermediate hop, the characteristics of maximum achievable goodput in a saturated condition will be different than in the case of direct communication. To demonstrate this, the two senarios were compared in a simple simulation.

Figure 5.4 (b) - '1 Source 1 Dest' displays the goodput measured in the 'ideal' scenarion as described earlier. It can be seen that the goodput remains fixed at the maximum achievable rate, with little deviation. Even though packets are dropped before transmission, the performance of the link (goodput and packet loss during transmission) is not affected.

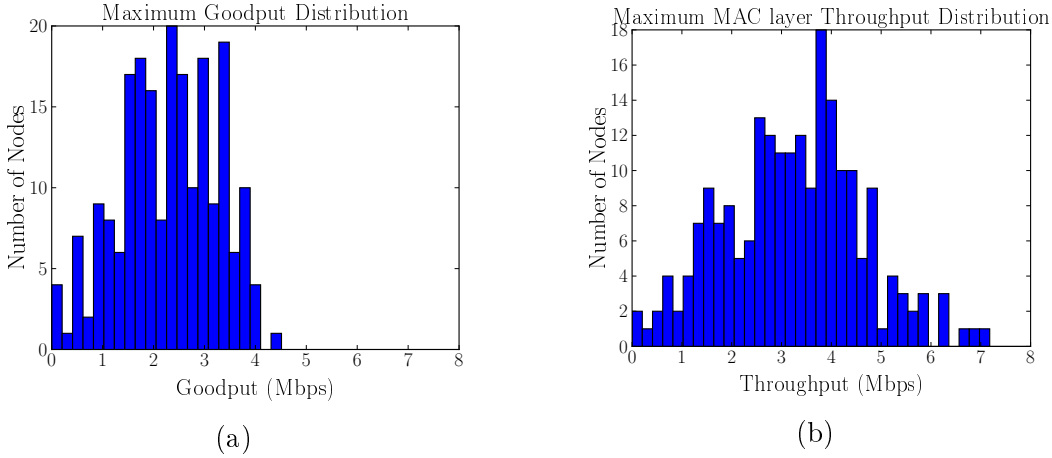
Figure 5.4 (b) - '3 Source 1 Hop 1 Dest' displays the goodput measured at the destination node when the intermediate node is in a saturated condition. Satuartion was caused by forcing the source nodes to send data at a rate greater than supported by the channel (i.e. the scenario depicted in (a)).

It can be seen that the goodput will have large fluctuations and a lower average value when compared to the direct communication scenario. In both

cases the link is saturated, but the characteristics vary. One can thus still deduce if a vehicular node was in a state of saturation by looking at the goodput or throughput over time on a per-node basis.

Figure 5.5 displays histograms of the maximum goodput and MAC layer throughput reached throughout the whole simulation period. It can be seen in (a) that the maximum goodput almost never exceeded 4 Mbps, thus never reaching a saturation point. Even when the MAC layer throughput is analysed in (b), it can be seen that only a few nodes reached values of around 7 Mbps, but never 8 Mbps or more. As seen in Figure 5.4 (b), goodput of more than 8 Mbps can be an indication that saturation occurred.

Figure 5.5: Distributions of maximum goodput and throughput per node



(a) (b)

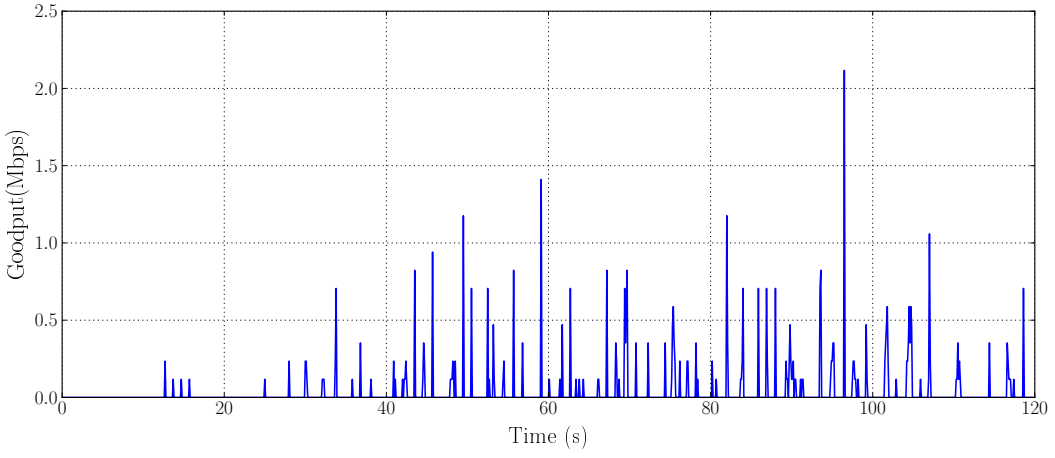


Figure 5.6: Goodput at aggregator at 35% penetration rate

Figure 5.6 displays the goodput measured at the aggregator at 35% penetration rate over the 120 second simulation time. The characteristics of the goodput data are almost impulse-like, with data being received in bursts rather than at a constant rate. This is attributed to the constant movement of nodes, with no fixed routes to the aggregator over time.

It can be seen that the aggregator was never near a point of saturation, with the maximum goodput achieved being just over 2 Mbps. When looking at MAC layer throughput, the same characteristics are observed with a maximum value of 5.6 Mbps. The same trends are observed across all other penetration rates.

Due to possible saturation only occurring at a small number of instances and never at the aggregator, it is concluded that saturation is not the main contributing cause of packets being lost in this scenario.

Link breakage and route discovery. Due to nodes in this scenario being highly mobile and traversing an urban area, constant route establishment and links being broken are a commonality. A single aggregator is used, and if a node is far away from the aggregator in terms of hop amount, a single link break in the route could be disastrous. If one link breaks in an active route, a new route needs to be created as per the operation of AODV. Creation of routes and notification of link breakages may cause increased control and management overhead in this scenario as discussed in Chapter 2.1.3.1 and Chapter 3.2.3.1. If the network is sparse and a new route to the aggregator cannot be established, data packets will be lost.

Figure 5.3 (d) indicates the average number of link breakages per node, and Figure 5.3 (c) indicates the average number of active routes in the route table for each node during its lifetime in the simulation. The amount of average active routes increase linearly as the penetration rate increases, which is to be expected since the amount of source nodes in the simulation increases linearly.

It is expected that an increase in routes per node, and network connectivity (or reach), will result in more link breakages per node. However, in Figure 5.3 (d) it can be seen that the average number of link breakages remains at a fairly constant, low level. At 30% penetration rate, each node has 42 average active routes with only about three link breakages on average. This is extremely promising, since it suggests that if a route is established it will most likely survive until the data transfer is complete, since the amount of data sent per node is minimal.

At low penetration rates of 10% and 15% the average number of link breaks is higher than observed at higher penetration rates. This can be attributed to the sparsity of nodes at lower penetration rates, resulting in nodes being in connection rage for shorter periods of time. The low number of link breakages at 5% penetration rate is attributed to the fact that nodes were extremely

Table 5.2: Overhead, dropped packets and data packet PER

Penetration Rate (%)	**5**	**10**	**15**	**20**	**25**	**30**	**35**
Avg Overhead (%)	16.84	27.74	29.83	36.53	43.23	44.74	42.17
% of All Packets Dropped due to **Full Queues**	0.00	3.58	5.94	5.44	8.75	12.39	11.94
% of All Packets Dropped due to **Bit Errors (PER %)**	2.91	9.09	13.43	17.48	26.34	22.36	26.74
Avg Data Packet PER (%)	0.20	4.70	5.90	10.20	13.10	12.10	10.90

sparse, and a much lower number of links were established in the first place. An increase in link breakages is seen from 25% to 35% penetration rate, and it is argued that interference, contention and larger links (in terms of hops) caused this increase.

An increase in the average number of routes with relatively constant number of average link breakages remaining relatively constant should theoretically be favourable and yield an increase in successful data transfer.

Therefore it is concluded that link breakages itself is not the cause of the low PDR in this scenario. Link breakages do not directly correlate with the observed PDR, and the ratio between active routes and link breakages remains relatively constant for all penetration rates.

Contention. Contention is the approach used to access a shared radio medium. In the case of IEEE 802.11p the EDCA mechanism is used for medium contention, detailed workings of which is explained in Section 2.1.1.1. In a scenario such as this simulation where a large number of nodes contend for the same medium, a node may struggle to gain medium access. If a large number of packets need to be sent or forwarded, the transmission queue can fill, up resulting in packets being dropped.

Table 5.2 presents, among other results, the percentage of all packets dropped (data, management and control) for the entire network over the 120 second simulation period. A very small percentage of packets are dropped due to full queues for penetration rates up to 20%. At higher penetration rates, this value increases to over 12%, which is a considerable amount of packets dropped.

As mentioned earlier, saturation is not the cause of full queues. It is argued that contention would mainly be responsible for the packets dropped due to full queues in this case. This makes sense when looking at the results in Table 5.2 - more active nodes are contending for medium access in higher penetration rates, causing queues to fill up due to contention resulting in dropped packets. This also explains the low percentage of packet drops at lower penetration rates, since sparse network density results fewer nodes competing for medium access in the same area.

Contention for medium access is concluded to be a contributing factor to the low PDR based on the presented argument and result.

Collisions. Collisions occur when multiple packets are received by a node at the same time. This will result in both packets being dropped at the PHY layer. This statistic is however not provided by INET.

Bit errors. Bit errors are bits in a data packet that were altered during transmission due to factors like interference, noise or synchronization errors. If these bits remain uncorrected, the packet is deemed corrupted and will be discarded. Error correction is typically performed by the transport layer, but since UDP is used in this scenario, no error correction is performed.

Forward error correction (FEC) is however performed at the PHY layer in IEEE 802.11, but there are no simulation statistics or results provided by INET regarding this process.

Table 5.2 displays the percentage of all packets in the network that were lost to bit errors (this includes data, management and control packets). At 5% and 10% penetration rates, only 2.91% and 9.09% packets are dropped due to bit errors. A steep increase in drop rate is seen up to 26.34% at 25% penetration rate, after which it remains fairly constant up to 35% penetration rate. This means that at higher penetration rates, on average, one out of every four packets received by a node will be corrupted due to interference from neighbouring nodes. At lower penetration rates, a very small number of bit errors are seen when compared to higher penetration rates, which is due to sparse node distribution.

The trend observed in this data has a direct inverse correlation with the PDR and LPDR curves seen in Figure 5.3 (a).

Table 5.2 also indicates the average packet error rate (PER) for data packets specifically. It can be seen that data PER is much lower than the overall PER, and remains around 10% to 13% from 20% to 35% penetration rate. This trend is due to a greater amount of management and control packets sent per data packet (i.e. overhead) at higher penetration rates. Table 5.2 indicates that the trend observed in the overall PER is reflected in the average network overhead percentage, confirming this assumption.

Due to the high bit error rates observed from 20% to 35% penetration rates, and the fact that the trend observed in the overall network bit error rate over penetration rates correlates with the data presented in Figure 5.3 (a), it is concluded that bit errors due to interference from neighbouring nodes is a direct cause of low PDR and LPDR values at higher penetration rates. However, at

lower penetration rates the cause of low PDR is not interference.

Physical positioning of nodes. Since all of the factors discussed so far does not seem to be major contributors to the low PDR (except bit errors), physical positioning of nodes (i.e. characteristics of vehicle mobility) has to be the culprit.

It is possible (especially at lower penetration rates) that a node or cluster of nodes are isolated for a period of time, i.e. not within communication range from other nodes as well as the aggregator. No route can be established to the aggregator in this scenario, and all packets generated during this time are regarded as lost / undeliverable. Isolation can also be caused by obstacles reducing signal power to the extent that no communication is possible even though nodes are theoretically within communication range.

At higher penetration rates, nodes are less sparse and the probability of a node or cluster being completely isolated becomes lower. The physical distance between nodes and the aggregator can, however, still be large, requiring many hops to reach the aggregator (since there is only one aggregator and it is not in a geographically central location). A larger number of hops would result in a higher probability of a link being broken or established in the first place (due to congestion and interference factors mentioned earlier).

At low penetration rates (5% and 10%) the combination of packets dropped due to full queues and bit errors in data packets combined only account for 0.2% and 8.28% of packets lost. However, the PDR was merely 18% and 17.02%. This means that physical isolation has to be the cause of 82% and 75% of packets lost respectively.

At higher penetration rates interference and contention starts to play a bigger role, causing a combined loss of between 22% and 25% of all data packets sent. Nodes that are further away from the aggregator may not be able to establish a link to it due to these factors. If these nodes were static, a link would eventually be established. But due to the vehicle's mobility characteristics, this task becomes difficult. This is reflected in the 70% packets lost that is not directly caused by any of the factors discussed earlier, and it is concluded that this loss has to be attributed to vehicle mobility characteristics.

In conclusion, at 5% and 10% penetration rates the loss of packets is almost entirely attributed to physical isolation of nodes. At higher penetration rates an increase in factors such as contention and interference is observed, causing a combined loss of between 22% and 25% of data packets. At these higher penetration rates the characteristics of vehicular movement and positioning are the major causes of lost packets, resulting in around 70% of data packets lost.

5.2.1.3 Suitability of IEEE 802.11p and AODV

IEEE 802.11p defines the PHY and MAC layer of the network stack, as discussed in Chapter 2. On the PHY layer it operates in 5.9 GHz band, which is theoretically more affected by interactions with physical objects than operation in the 2.4 GHz band.

Contention and interference has a large impact on network performance in this scenario as discussed earlier. If a IEEE 802.11 standard operating in the 2.4 GHz band was to be used instead of IEEE 802.11p, it is assumed that contention and interference levels would greatly increase. Many modern consumer electronic devices operate in the 2.4 GHz band, and if it is assumed that each vehicle (active nodes and standard vehicles) has at least one or more such devices, a potentially large degradation in performance would be observed.

It is concluded that operation in the 5.9 GHz band is more optimal than 2.4 GHz in the case of this scenario, specifically when looking at potential contention and interference levels.

Utilising channel switching as featured in IEEE 1609 WAVE and ETSI ITS-G5 could result in lower interference and contention, however, in these standards a single non-safety application does not utilise multiple channels. In this regard the sole use of IEEE 802.11p would be just as effective, if not more effective, than the abovementioned standards. In both of these standards a non-safety application would only be able to communicate in 50% of the time since the control channels will take priority, as mentioned in Section 2.1.1.

Due to saturation rarely being encountered, as seen in Figure 5.5 (a) and (b), the lower bit rates of IEEE 802.11p in this scenario are still suitable to transmit the amount of data generated.

Network overhead is quite substantial in this application scenario, as seen in Table 5.2. With 32 nodes present in the simulation (5% penetration rate), an average overhead of 16.84% is observed. This increases to a maximum 44.74% at 30% penetration rate.

The ratio of link breakages to active routes is also low, with an average of around one link breakage for every 15 routes established with 147 nodes in the simulation (25% penetration rate). At 209 nodes (35% penetration rate) the lowest value of around one link breakage for every 8 routes established, which is still favourable. It is concluded that route discovery is the main cause of the high network overhead in this case.

Average network connectivity is between 22% and 25% across all penetration rates, as displayed in Table 5.3. For this specific application and scenario no data is stored by nodes after attempting to send the data, and no data is shared

Table 5.3: Average network connectivity

Penetration rate (%)	**5**	**10**	**15**	**20**	**25**	**30**	**35**
Network connectivity (%)	24.84	25.08	22.94	24.07	23.90	23.01	22.19

with neighbouring nodes with the intention that they store the data until a link to the aggregator is possible. It is assumed that for this case, routing protocols categorized as geocast, broadcast or cluster-based would not be able to achieve similar connectivity levels without causing a large increase in the network overhead. As seen in Figure 5.5 (b), this may result in more nodes reaching a point of saturation, causing more packets to be dropped. Pure location-based protocols, as discussed in Chapter 2, is also assumed to not be able to outperform AODV in this case, due to the large impact of obstacles on connectivity, as seen and discussed in Chapter 4.

Other topology-based routing protocols were not directly compared to AODV in this work, but based on results in [86] other topology-based routing protocols may yield similar results when compared to AODV.

Based on these arguments it is concluded that topology-based routing protocols, and specifically AODV in this case, is suitable for decentralised license plate detection (as implemented in this scenario) compared to other existing ad hoc routing protocols.

5.3 Summary

In this chapter, an application implementation and simulation scenario was clearly defined to realise the envisaged solution. Large scale simulations were performed within the verified and calibrated simulation environment over various active node penetration rates. Application level data generation was investigated and it was concluded that the implemented technologies would be more than capable of transporting the data.

Network performance in the given scenario was investigated and suitability of IEEE 802.11p and AODV for use in decentralised licence plate detection was determined. It was concluded that low PDR can be attributed to physical node isolation at lower penetration rates and to vehicle mobility characteristics, congestion and interference at higher penetration rates.

Based on results and arguments stipulated in this chapter, it was concluded that IEEE 802.11p and AODV are suitable for decentralised licence plate detection and reporting in this specific scenario. There are, however, possible optimisations and improvements to be made to the higher networking layers such as transport and application layers, which could possibly yield a large

performance increase in terms of data deliverability.

Chapter 6

Conclusions

6.1 Conclusion

This project set out to investigate the implementation and evaluation of existing VANET standards and protocols for suitability of distributed licence plate detection and reporting in an urban scenario. Large scale deployment of existing hardware solutions was not possible since it would cost approximately ZAR 320 000 to deploy 25 active nodes with commercially available hardware. Since the budget was set at ZAR 15 000, a simulation approach was used to address the problem.

1) Realising a simulation environment with realistic wireless networking and vehicular traffic aspects with which to evaluate VANET standards and protocols for suitability of usage in the envisaged solution.

South Africa does not have spectrum allocation for ITS applications, and it was assumed that South Africa will follow the spectrum allocation trends of Europe. Frequencies utilised by ETSI ITS-G5 and IEEE 1609 WAVE fall within the spectrum allocation reserved for ITS applications, ruling out AIRB STD-T109 as an usable standard in this project since it operates in the 700 MHz band.

It was stated that the proposed solution would be a standalone application and that it will not be integrated with existing ITS infrastructure and systems, thus the implementation of either ITS-G5 or IEEE 1609 is not required. It was also deemed possible to create the envisaged solution solely with IEEE 802.11p without relying on the upper layer network specifications provided by the aforementioned network standards. Due to these assumptions and the assumption that it would be difficult to obtain comprehensive, fully working open-source implementations of ITS-G5 or IEEE 1609 WAVE, IEEE 802.11p alone was chosen as the base network standard for this project.

A simulation environment with realistic wireless networking and vehicular traffic patterns with which to evaluate VANET standards and protocols for suitability of usage in the envisaged solution was successfully realised.

OMNeT++, INET, Veins_INET and SUMO were the simulation tools chosen. INET provides models for IEEE 802.11p, signal propagation effects, the physical environment as well as various ad hoc routing protocols, meeting the requirements of this project. OMNeT++ provides scalability and comprehensive result analysis, whilst integration of vehicle mobility simulation with INET is possible within OMNeT++ by using Veins_INET and SUMO. SUMO provides the option of accurate vehicle traffic simulation as well as the creation of custom maps on which to simulate vehicle traffic, meeting the specified requirements set forth.

Setting up the network simulation started by configuring PHY and MAC layer parameters in INET to comply with the IEEE 802.11p specification. An operating frequency of 5.86 GHz was chosen to comply with the non-safety related channel allocation set forth by ITS-G5 and IEEE 1609. The implementation was briefly tested by simulation of a simple scenario. A throughput result of 4.6 Mbps matched that of related research, confirming correct implementation.

Signal fading models in INET were compared and the NakagamiFading model was selected for use. It provided the most aggressive path loss model and literature stated that this model more accurately matches empirical data when compared to the other INET models.

AODV was selected as the routing protocol of choice. It was chosen due to it being the most well-known and widely used routing protocol out of the ad hoc routing protocols available in INET based on literature appearances, as well as being fully implemented in INET according to RFC 3561. Open-source implementations of AODV is available and easily obtainable, which provides more motivation for choosing AODV since a hardware solution has to be created based on the simulation environment.

INET and SUMO were configured to run in parallel by using Veins_INET as the module which controls the information transfer of vehicle positional data between the two simulation tools. Correct functionality and integration were confirmed by visually inspecting vehicle positions in INET at specific times, and comparing it to an identical simulation scenario implemented with the Veins framework.

Central Stellenbosch map data from OpenStreetMap was converted to a SUMO road network map which was used as the area for traffic simulation. Adjustments were made to the road network map in order to eliminate unwanted areas and to correct errors induced during the conversion process in order to make sure vehicles travelled only in the urban core of Stellenbosch.

Physical obstructions (obstacles) were implemented by converting the appropriate map data from OpenStreetMap to a SUMO polygon description file, and from there to a format understandable by INET. Correct placement of obstacles in relation to the SUMO road map was verified by comparing the resulting positioning of obstacles to that of the Veins framework, which uses the same approach and is known to be accurate. All obstacles were assumed to be of homogeneous material and equal height. INET's physical environment material properties were investigated and concrete was chosen due to it having the most aggressive impact on signal power.

In conclusion, a network simulation environment implementing IEEE 802.11p, AODV and path loss models was integrated with a vehicle traffic simulation environment which simulates traffic on a road network map of central Stellenbosch. Physical obstructions were also added to the network simulation in correct relative positioning to the road network map. This simulation environment was capable of evaluating IEEE 802.11p and AODV for suitability of realising the envisaged solution, but needed to be verified and calibrated to ensure that it provided a realistic approximation of expected real-world results.

2) Verification and calibration of the simulation environment by means of a hardware solution.

A hardware solution was needed to verify and calibrate the networking simulation environment and to ensure that it provides an accurate approximation of a real-world scenario.

A HP AR5BHB92-H mini-PCIe wireless adapter with an Atheros AR9280 chipset was chosen for the implementation of the PHY layer. This hardware was chosen because it uses open-source ath9k drivers in Linux and is capable of operation in the 5.9 GHz band. 2 dBi omnidirectional whip antennas were used. For the compute platform, Proline NT425 single board computers were used, kindly offered by Telectro CC. These computers met the requirements set forth in Chapter 2, and since obtaining these single board computers at no cost, no other options were investigated.

Implementing IEEE 802.11p was achieved by making use of the '802.11p-linux' project by Czech Technical University, which provided changes to mac-80211 (and other related kernel modules) to fully implement IEEE 802.11p. Ubuntu 15.04 (kernel version 3.19) was the operating system of choice since it was the first mainline kernel to release with MAC layer modifications proposed by the '802.11p-linux' project. Changes to the ath9k driver and regulatory rules were made to make the implementation compliant with IEEE 802.11p.

AODV-UU was chosen for AODV implementation due to it being the latest up-to-date open-source AODV implementation readily available. Modifica-

tions were made to the source code to make it compatible with Linux kernels up to version 3.19.8, since it only supports kernel versions 2.4.x and 2.6.x. The updated AODV-UU was compared to an implementation of the original version and successful, identical functionality was obtained.

Hardware experimentation was performed with the created hardware solution to successfully verify and configure core aspects of the network simulation environment. The default PHY configuration parameters of the hardware implementation was finalised. The TX power of the NIC in OCB configuration is 17 dbm, with an estimated RX sensitivity of -85 dbm. An operating frequency of 5.86 GHz was selected as per the reasoning mentioned during simulation setup. The data bit rate had to be determined by means of experimentation. Wireshark, iperf and a custom UDP application created to superseed iperf were used in all hardware experiments. The UDP application was verified to yield the same results as iperf with identical configurations, by sending UDP data with 1470 B payload to a second node at 1 Gbps to force a saturated condition. With the PHY hardware configuration at default values, iperf yielded an average goodput of 8.47 Mbps, while the custom UDP application yielded 8.41 Mbps, confirming correct functionality of the UDP application. By making use of this application, the maximum achievable goodput at various payload lengths was measured in order to determine the default data rate of the hardware solution. Experimental results were compared to theoretical calculations and simulation results, resulting in the conclusion that a data rate of 12 Mbps is being used. Trends in simulation results at 12 Mbps data rate matched the experimental data, with a maximum difference of 5.56% and a most optimal point at 1470B (the MTU of Ethernet V2) yielding a difference of 2.27%. These results validated the simulation environment's PHY layer implementation. A payload of 1470 B was thus used in further experiments and simulations.

Experimentation to observe medium contention was performed by having two nodes simultaneously sending data to a single node at rates greater than 12 Mbps to force saturation. Trends observed in the empirical data matched simulation results of an identical scenario, validating the MAC layer implementation in this scenario.

Signal fading implementation was verified by measuring maximum goodput over distance between two hardware nodes. Performance characteristics were very similar between simulation and empirical data up to a distance of 220m, after which the simulation yielded an overestimation of performance. Simulation parameters were adjusted to present a more pessimistic view. The negative effects of antenna polarisation and radiation patterns on achievable performance at increasing distances were observed during the hardware experimentation. The choice of antenna was deemed to be a crucial factor when aiming to deploy the solution in the real world.

The effect of signal reflection from metallic objects (parked vehicles) was observed to be fairly significant. It was also noticed that when a vehicle breaks the line of sight between two nodes, it completely breaks the link. These observations highlighted the potential severe negative effects that vehicles will have on an actual VANET deployment. These effects were, however, not implemented in this project's simulation environment, but should be an important addition to a future iteration of the simulation environment.

Implementation of physical obstructions in simulation was compared to empirical data obtained through experimentation in identical scenarios. Material properties of buildings in the simulation were adjusted such that it provides a more accurate representation of the empirical data.

In conclusion, a hardware solution fully implementing IEEE 802.11p (PHY and MAC) was successfully created by using a single board computer, wireless mini-PCIe adapter with an Atheros A9280 chipset and a custom Linux kernel implementing an adjusted MAC layer and device drivers. AODV-UU was modified to operate on the same kernel version, resulting in a 802.11p-enabled hardware solution capable of multi-hop data transfer. This hardware solution was used to successfully verify and calibrate core aspects of the network simulation environment by means of experimentation in the field.

This solution can be used by other researchers in the field in need of an affordable, open-source implementation of a VANET OBU. Source code of the modified AODV-UU implementation was made publicly available for any researcher in need of implementing AODV-UU on a Linux kernel version newer than 2.6.

3) Execution of large-scale simulations with the calibrated and verified simulation environment to extrapolate what the performance of the envisaged solution would be within the scope of the specified scenario, and to determine suitability of the chosen networking standards and protocols for this specific scenario.

An application implementation and a simulation scenario were clearly defined and successfully set up, to realise the 'envisaged solution' within the verified and calibrated simulation environment created throughout Chapter 3 and Chapter 4. Central Stellenbosch was used as the geographical location for vehicle traffic simulation and obstacle creation, as discussed in Chapter 3. It was decided to place a single aggregator in the simulation environment, on the intersection of Merriman Ave and Bird St. This location was chosen due to it being the point of maximum traffic throughput, determined by knowledge of the area as well as inspection in the SUMO simulation. Simulation time limit of 120 seconds was used since it provided suitable real-world simulation execution times.

A simple licence plate detection and reporting application was defined and created, and successfully implemented in the discussed simulation scenario. The application is run on each active node, and evaluates the simulation environment 10 times per second. If a vehicle is detected within a Euclidean distance of 20 m from the node, its information is logged. After 10 detection cycles, the data is sent to the aggregator in bursts of UDP packets with 1470 B payload. Based on assumptions discussed in Section 5.1.1, 29 detections can fit into a single payload.

The application was evaluated in the simulation environment at active node penetration rates of 5% to 35%. 11560 vehicle detections were made and 839 data packets generated at 5% penetration rate. At 35% penetration rate, 84279 vehicle detections were made with 6020 data packets generated. An average of 14 detections' metadata were contained in each packet's payload, irrespective of penetration rate and amount of detections. This means that it is possible to double the amount of detection data sent without resulting in any additional overhead or impact on the network performance.

It was observed that 20% of the active nodes generated 60% of the data, which is attributed to vehicular traffic characteristics. This may cause congestion or saturation at certain points in the network.

Overall, in the case of 35% penetration rate (209 active nodes) a total of 4.21 MB data was generated during the 120 second simulation duration. Assuming 24 hour operation, merely 3.03 GB data will be generated per day (14.5 MB per node). This drops to 416.16 MB for 32 active nodes, for an average of 13 MB per node per day. This amount of data is miniscule by the standard of modern day technologies, and the network technologies implemented in this scenario as well as related technologies would be more than capable of transporting this amount of data.

However, a maximum PDR of 18% was measured at 5% penetration rate with the PDR declining as penetration rate increases, reaching a steady state of between 6% and 7% at 25% to 35% penetration rate. These values are extremely low given the fact that the network is not taxed with large amounts of data. It was concluded that physically isolated nodes or node clusters were the cause of lost packets at 5% and 10% penetration rates, with 97.6% and 90% of packets lost due to this factor. Physical isolation is caused by the characteristics of node positional and mobility characteristics, i.e. the way in which traffic flows and vehicles behave in this scenario. Storing data and transmitting it only when medium becomes available is a possible solution for this issue.

At higher penetration rates, contention and bit errors caused by interference were identified as factors contributing to the low PDR, which caused the loss of between 22% and 25% of packets. Vehicular positional and mobility char-

acteristics are still the direct cause of the rest of the lost packets at higher penetration rates.

It was concluded that IEEE 802.11p's operation in the 5.9 GHz band is more optimal than using 2.4 GHz in this scenario, since many modern consumer electronic devices operate in the 2.4 GHz band. It is argued that by using the 2.4 GHz band, this solution would only increase contention and interference, resulting in worse performance. Thus 5.9 GHz is more suitable in this scenario. Utilising channel switching as featured in IEEE 1609 WAVE and ETSI ITS-G5 could result in lower interference and contention, however, the non-safety application in this scenario would only be able to send and receive data during 50% of the transmission slots. In this respect it is argued that the sole use of IEEE 802.11p would be just as effective, if not more effective, than the above mentioned standards. It was shown that saturation was rarely encountered, with only three nodes reaching a potential point of saturation at 35% penetration rate. Thus, the lower bit rates of IEEE 802.11p are still suitable for this scenario to transmit the amount of generated data.

Based on these results and arguments, it was concluded that IEEE 802.11p is suitable for use in decentralised licence plate detection and reporting in this specific simulation scenario. Average network overhead of 16.84% at 5% penetration rate was observed increasing to a maximum of 44.74% at 30% penetration rate. From 25% to 35% the average overhead stabilised to values of around 42 - 44%.

The ratio of link breakages to active routes displayed an average of around one link breakage for every fifteen routes established at 25% penetration rate. 35% penetration rate had the lowest breakage of one link for every eight active routes. It was concluded that route discovery is the main cause of the high network overhead.

Average network connectivity was constantly between 22% and 25% across all penetration rates. It is assumed that for this case, routing protocols categorized as geocast, broadcast, cluster-based or location-based would not be able to achieve similar connectivity levels without causing a large increase in the network overhead, resulting in more packets dropped due to possible link saturation.

Based on these arguments, it is concluded that topology-based routing protocols, and specifically AODV in this case, are suitable for decentralised license plate detection as implemented in this scenario when compared to other existing ad hoc routing protocols.

6.2 Recommendations

Antenna characteristics are considerations in designing a VANET OBU. It was concluded during experimentation that antenna characteristics such as radiation pattern will greatly affect communication reliability in mobile node conditions, especially at large distances such as above 200 m. The importance of antenna choice was underestimated when the hardware solution was created in this project, resulting in a solution that may have periods of unreliable performance due to device orientation and rotation.

It is recommended that future researchers aiming to create a custom VANET OBU, should factor in the importance of antenna selection and design.

6.3 Future Work

Signal reflection caused by surrounding vehicles was observed to have a significant negative impact on wireless communication. Experimentation was performed in a small street scenario, and it is assumed that this effect will be even larger in congested traffic scenarios (commonly found in urban areas). It is envisioned that this effect would be thoroughly investigated and quantified in the future, and that physical vehicle models could be implemented into the simulation in pursuit of a more accurate simulation scenario.

Performing field tests with all five or more nodes will be the next step in assessing the feasibility of the envisaged solution. Nodes can be placed inside vehicles and a test scenario can be executed and compared with a similar scenario in simulation. The impact on communication and connectivity when nodes are placed inside vehicles (essentially a Faraday cage) can also be investigated and quantified.

Intelligent data management can be implemented to address issues such as packet loss due to no link to destination. Data can for example be stored for a certain period of time when no link is established, instead of attempting and failing to send the data. Important data can be shared with and stored by neighbouring nodes to increase the chance of delivery. Additional features such as beaconing (such as implemented by IEEE 1609) can also be added to potentially aid route establishment.

Data privacy and security did not receive any attention in this project, but will be a significant topic when attempting to deploy the solution as a product or service. Aspects of legality, ethics as well as technical challenges regarding implementation of data privacy and security should be investigated in the future.

List of References

[1] OSI model [Online]: Available at: `https://en.wikipedia.org/wiki/OSI{_}model`, [2018, February 15], 2018.

[2] Grafling, S., Mahonen, P. and Riihijarvi, J.: Performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications. In: *2010 Second International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 344–348. IEEE, 2010. ISBN 978-1-4244-8088-3.

[3] Fernandes, B., Rufino, J., Alam, M. and Ferreira, J.: Implementation and Analysis of IEEE and ETSI Security Standards for Vehicular Communications. *Mobile Networks and Applications*, pp. 1–10, 2018. ISSN 1383-469X.

[4] Heinovski, J., Klingler, F., Dressler, F. and Sommer, C.: Performance comparison of IEEE 802.11p and ARIB STD-T109. In: *2016 IEEE Vehicular Networking Conference (VNC)*, pp. 1–8. IEEE, 2016. ISBN 978-1-5090-5197-7.

[5] Tareq, M., Alam, D.A., Islam, M. and Ahmed, R.: Simple half-wave dipole antenna analysis for wireless applications by cst microwave studio. *International Journal of Computer Applications*, vol. 94, no. 7, 2014.

[6] RFA-02-L2H1 [Online]: Available at: `https://www.mouser.co.za/datasheet/2/268/microchip_RFA-02-L2H1-1181314.pdf`, [2018].

[7] Ieee standard for information technology– local and metropolitan area networks– specific requirements– part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 6: Wireless access in vehicular environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)*, pp. 1–51, July 2010.

[8] Ieee standard for information technology–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications - amendment 8: Medium access control (mac) quality of service enhancements. *IEEE Std 802.11e-2005 (Amendment to IEEE Std 802.11, 1999 Edition (Reaff 2003)*, pp. 1–212, 2005.

[9] ETSI: Intelligent transport systems (its); access layer specification for intelligent transport systems operating in the 5 ghz frequency band. Standard EN 302 663 V1.2.1, ETSI, 2013.

[10] Crime situation in South Africa[Online]: Available at: `https://www.saps.gov.za/services/final-crime-stats-release-02september2016.pdf`, [2018, September 28], 2016.

[11] Du, S., Ibrahim, M., Shehata, M. and Badawy, W.: Automatic license plate recognition (alpr): A state-of-the-art review. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, Feb 2013. ISSN 1051-8215.

[12] Alegria, F. and Girao, P.S.: Vehicle plate recognition for wireless traffic control and law enforcement system. In: *2006 IEEE International Conference on Industrial Technology*, pp. 1800–1804. Dec 2006.

[13] Hsu, G., Chen, J. and Chung, Y.: Application-oriented license plate recognition. *IEEE Transactions on Vehicular Technology*, vol. 62, no. 2, pp. 552–561, Feb 2013. ISSN 0018-9545.

[14] MVTRAC [Online]: Available at: `https://www.mvtrac.com/alpr`, [2017, March 17], 2015.

[15] Mobile ALPR [Online]: Available at: `https://roadmetric.com/solution/mobile-anpr-lpr/`, [2017, March 17], 2017.

[16] Department of Homeland Security cancels national license-plate tracking plan [Online]: Available at: `https://www.washingtonpost.com/world/national-security/dhs-cancels-national-license-plate-tracking-plan/2014/02/19/a4c3ef2e-99b4-11e3-b931-0204122c514b_story.html?noredirect=on&utm_term=.cfc69b7857f3`, [2017, March 17], 2014.

[17] Eze, E.C., Zhang, S. and Liu, E.: Vehicular ad hoc networks (VANETs): Current state, challenges, potentials and way forward. In: *2014 20th International Conference on Automation and Computing*, pp. 176–181. IEEE, sep 2014. ISBN 978-1-9095-2202-2.
Available at: `http://ieeexplore.ieee.org/document/6935482/`

[18] Papadimitratos, P., La Fortelle, A., Evenssen, K., Brignolo, R. and Cosenza, S.: Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine*, vol. 47, no. 11, pp. 84–95, nov 2009. ISSN 0163-6804.
Available at: `http://ieeexplore.ieee.org/document/5307471/`

[19] IEEE Guide for Wireless Access in Vehicular Environments (WAVE) - Architecture. *IEEE Std 1609.0-2013*, pp. 1–78, March 2014.

[20] ETSI: EN 302 665 - V1.1.1 - Intelligent Transport Systems (ITS); Communications Architecture. Tech. Rep., 2010.
Available at: `http://www.etsi.org/deliver/etsi{_}en/302600{_}302699/302665/01.01.01{_}60/en{_}302665v010101p.pdf`

[21] ARIB: 700 MHz Band Intelligent Transport Systems ARIB Standard. 2012. Available at: `http://www.arib.or.jp/english/html/overview/doc/5-STD-T109v1_1-E1.pdf`

[22] FCC Allocates Spectrum in 5.9 GHz Range for Intelligent Transportation Systems Uses Action Will Improve the Efficiency of the Nation's Transportation Infrastructure. 1999. Available at: `https://apps.fcc.gov/edocs{_}public/attachmatch/DOC-177370A1.pdf`

[23] Sommer, C.: *Vehicular networking*. Cambridge University Press, 2014.

[24] Fernandez, J.A., Borries, K., Cheng, L., Vijaya Kumar, B.V.K., Stancil, D.D. and Bai, F.: Performance of the 802.11p Physical Layer in Vehicle-to-Vehicle Environments. *IEEE Transactions on Vehicular Technology*, vol. 61, no. 1, pp. 3–14, jan 2012. ISSN 0018-9545.

[25] *IEEE trial-use standard for wireless access in vehicular environments (WAVE)–resource manager*. Institute of Electrical and Electronics Engineers, 2006. ISBN 9780738152424.

[26] IEEE Standard for Wireless Access in Vehicular Environments Security Services for Applications and Management Messages. *IEEE Std 1609.2-2013 (Revision of IEEE Std 1609.2-2006)*, pp. 1–289, April 2013.

[27] IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services. *IEEE Std 1609.3-2010 (Revision of IEEE Std 1609.3-2007)*, pp. 1–144, Dec 2010.

[28] IEEE Standard for Wireless Access in Vehicular Environments (WAVE)–Multichannel Operation. *IEEE Std 1609.4-2010 (Revision of IEEE Std 1609.4-2006)*, pp. 1–89, Feb 2011.

[29] ETSI: TR 101 607 - V1.1.1 - Intelligent Transport Systems (ITS); Cooperative ITS (C-ITS); Release 1. Tech. Rep., . Available at: `http://www.etsi.org/deliver/etsi{_}tr/101600{_}101699/101607/01.01.01{_}60/tr{_}101607v010101p.pdf`

[30] ETSI: TS 103 175 - V1.1.1 - Intelligent Transport Systems (ITS); Cross Layer DCC Management Entity for operation in the ITS G5A and ITS G5B medium. Tech. Rep., . Available at: `http://www.etsi.org/deliver/etsi{_}ts/103100{_}103199/103175/01.01.01{_}60/ts{_}103175v010101p.pdf`

[31] ETSI: TS 102 636-4-1 - V1.1.1 - Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality. Tech. Rep., . Available at: `http://www.etsi.org/deliver/etsi{_}ts/102600{_}102699/1026360401/01.01.01{_}60/ts{_}1026360401v010101p.pdf`

[32] ETSI: TS 102 636-5-1 - V1.1.1 - Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol. Tech. Rep., 2011.
Available at: http://www.etsi.org/deliver/etsi{_}ts/102600{_}102699/1026360501/01.01.01{_}60/ts{_}1026360501v010101p.pdf

[33] ETSI: TS 102 637-1 - V1.1.1 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements. Tech. Rep., 2010.
Available at: http://portal.etsi.org/chaircor/ETSI{_}support.asp

[34] ETSI: TS 102 637-2 - V1.2.1 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. Tech. Rep., 2010.
Available at: http://www.etsi.orghttp://portal.etsi.org/tb/status/status.asphttp://portal.etsi.org/chaircor/ETSI{_}support.asp

[35] ETSI: TS 102 637-3 - V1.1.1 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service. Tech. Rep., 2010.
Available at: http://www.etsi.orghttp://portal.etsi.org/tb/status/status.asphttp://portal.etsi.org/chaircor/ETSI{_}support.asp

[36] Document 32008D0671 [Online]: Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32008D0671, [2017].

[37] ICASA NOTICE 1442 OF 2004 [Online]: Available at: https://www.gov.za/sites/default/files/gcis_document/201409/265840.pdf, [2017].

[38] ECA Table [Online]: Available at: https://www.efis.dk/sitecontent.jsp?sitecontent=ecatable, [2017].

[39] Medhi, D. and Ramasamy, K.: *Network Routing: Algorithms, Protocols, and Architectures (The Morgan Kaufmann Series in Networking)*. Morgan Kaufmann, 2007. ISBN 0120885883.

[40] Sharef, B.T., Alsaqour, R.A. and Ismail, M.: Vehicular communication ad hoc routing protocols: A survey. *Journal of Network and Computer Applications*, vol. 40, pp. 363–396, apr 2014. ISSN 1084-8045.
Available at: https://www-sciencedirect-com.ez.sun.ac.za/science/article/pii/S1084804513001963?via{%}3Dihub

[41] Li, F. and Wang, Y.: Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular Technology Magazine*, vol. 2, no. 2, pp. 12–22, 2007. ISSN 1556-6072.
Available at: http://ieeexplore.ieee.org/document/4450627/

[42] Singh, S. and Agrawal, S.: VANET routing protocols: Issues and challenges. In: *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1–5. IEEE, mar 2014. ISBN 978-1-4799-2291-8.
Available at: http://ieeexplore.ieee.org/document/6799625/

[43] Härri, J., Filali, F. and Bonnet, C.: Mobility Models for Vehicular Ad Hoc Networks: A Survey and Taxonomy. 2006.
Available at: `http://www.eurecom.fr/en/publication/1951/download/cm-haerje-060320.pdf`

[44] Burghout, W., Koutsopoulos, H. and Andreasson, I.: A discrete-event mesoscopic traffic simulation model for hybrid traffic simulation. In: *2006 IEEE Intelligent Transportation Systems Conference*, pp. 1102–1107. IEEE, 2006. ISBN 1-4244-0093-7.
Available at: `http://ieeexplore.ieee.org/document/1707369/`

[45] Grzybek, A., Seredynski, M., Danoy, G. and Bouvry, P.: Aspects and trends in realistic VANET simulations. In: *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–6. IEEE, jun 2012. ISBN 978-1-4673-1239-4.
Available at: `http://ieeexplore.ieee.org/document/6263793/`

[46] Krauss, S., Wagner, P. and Gawron, C.: Metastable states in a microscopic model of traffic flow. *Physical Review E*, vol. 55, no. 5, pp. 5597–5602, may 1997. ISSN 1063-651X.
Available at: `https://link.aps.org/doi/10.1103/PhysRevE.55.5597`

[47] Gadkari, M.Y. and Sambre, N.B.: VANET: Routing Protocols, Security Issues and Simulation Tools. *IOSR Journal of Computer Engineering*, vol. 3, no. 3, pp. 2278–661.
Available at: `www.iosrjournals.org`

[48] A survey and comparative study of simulators for vehicular ad hoc networks (VANETs). *Wireless Communications and Mobile Computing*, vol. 11, no. 7, pp. 813–828, jul 2011.
Available at: `http://doi.wiley.com/10.1002/wcm.859`

[49] The Network Simulator - ns-2[Online]: Available at: `https://www.isi.edu/nsnam/ns/`, [2017, March 11].

[50] ns-3 | a discrete-event network simulator for internet systems[Online]: Available at: `https://www.nsnam.org/`, [2017, March 11], 2017.

[51] OMNeT++[Online]: Available at: `https://www.omnetpp.org/intro`, [2017, March 11], 2017.

[52] Veins [Online]: Available at: `http://veins.car2x.org/`, [2017, March 11], 2017.

[53] INET Framework [Online]: Available at: `https://inet.omnetpp.org/`, [2017, March 11], 2017.

[54] Zeng, X., Bagrodia, R. and Gerla, M.: GloMoSim: a library for parallel simulation of large-scale wireless networks. In: *Proceedings. Twelfth Workshop on*

Parallel and Distributed Simulation PADS '98 (Cat. No.98TB100233), pp. 154–161. IEEE Comput. Soc, 1998. ISBN 0-8186-8457-7.
Available at: `http://ieeexplore.ieee.org/document/685281/`

[55] Bagrodia, R., Meyer, R., Takai, M., Yu-An Chen, Xiang Zeng, Martin, J. and Ha Yoon Song: Parsec: a parallel simulation environment for complex systems. vol. 31, pp. 77–85. 1998. ISSN 00189162.
Available at: `http://ieeexplore.ieee.org/document/722293/`

[56] QualNet Network Simulator Software [Online]: Available at: `https://web.scalable-networks.com/qualnet-network-simulator-software`, [2017, March 11], 2017.

[57] NetSim [Online]: Available at: `https://www.tetcos.com/index.html`, [2017, March 11], 2017.

[58] OPNET Network Simulator [Online]: Available at: `http://opnetprojects.com/opnet-network-simulator/`, [2017, March 11], 2017.

[59] Hogie, L., Bouvry, P. and Guinand, F.: An Overview of MANETs Simulation. *Electronic Notes in Theoretical Computer Science*, vol. 150, no. 1, pp. 81–101, mar 2006. ISSN 1571-0661.
Available at: `https://www-sciencedirect-com.ez.sun.ac.za/science/article/pii/S1571066106001010?via{%}3Dihub`

[60] Weingartner, E., vom Lehn, H. and Wehrle, K.: A Performance Comparison of Recent Network Simulators. In: *2009 IEEE International Conference on Communications*, pp. 1–5. IEEE, jun 2009.
Available at: `http://ieeexplore.ieee.org/document/5198657/`

[61] WaveCombo Solutions [Online]: Available at: `http://www.redpinesignals.com/Products/802.11p_V2X_Connectivity/WAVECombo_Solution.php`, [2017, March 15], 2017.

[62] LocoMate Products [Online]: Available at: `https://www.aradasystems.com/products/`, [2017, March 15], 2017.

[63] Cohda Wireless Hardware [Online]: Available at: `https://cohdawireless.com/solutions/hardware/`, [2017, March 15], 2017.

[64] Theo P173 Module [Online]: Available at: `https://www.u-blox.com/en/product/theo-p173-module`, [2017, March 15], 2017.

[65] Vera P1 Series [Online]: Available at: `https://www.u-blox.com/en/product/vera-p1-series`, [2017, March 15], 2017.

[66] González, V., Santos, A.L., Pinart, C. and Milagro, F.: Experimental Demonstration of the Viability of IEEE 802.11b Based Inter-vehicle Communications. In: *Proceedings of the 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, TridentCom

'08, pp. 1:1—-1:7. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 2008. ISBN 978-963-9799-24-0.

[67] Santa, J., Tsukada, M., Ernst, T. and Gomez-Skarmeta, A.F.: Experimental Analysis of Multi-hop Routing in Vehicular Ad-hoc Networks. In: *TridentCom '08 Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities*, pp. —-. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) ICST, Brussels, Belgium, Belgium ©2008, 2009.

[68] Agafonovs, N., Strazdins, G. and Greitans, M.: Accessible, customizable, high-performance IEEE 802.11p vehicular communication solution. In: *2012 The 11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, pp. 127–132. IEEE, jun 2012. ISBN 978-1-4673-2039-9.

[69] Barcelos, V.P., Amarante, T.C., Drury, C.D. and Correia, L.H.A.: Vehicle monitoring system using IEEE 802.11p device and Android application. In: *2014 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–7. IEEE, jun 2014. ISBN 978-1-4799-4277-0.

[70] Vivek, N., Sowjanya, P., Sunny, B. and Srikanth, S.V.: Implementation of IEEE 1609 WAVE/DSRC stack in Linux. In: *TENSYMP 2017 - IEEE International Symposium on Technologies for Smart Cities*, pp. 1–5. IEEE, jul 2017. ISBN 9781509062553.

[71] Abunei, A., Comsa, C.-R. and Bogdan, I.: Implementation of a Cost-effective V2X hardware and software platform. In: *2016 International Conference on Communications (COMM)*, pp. 367–370. IEEE, jun 2016. ISBN 978-1-4673-8197-0.

[72] en:users:drivers:ath5k [Linux Wireless] [Online]: Available at: `https://wireless.wiki.kernel.org/en/users/drivers/ath5k`, [2018, February 20], 2017.

[73] en:users:drivers:ath9k [Linux Wireless] [Online]: Available at: `https://wireless.wiki.kernel.org/en/users/drivers/ath9k`, [2018, February 20], 2017.

[74] Supported Atheros wireless devices [Online]: Available at: `https://wireless.wiki.kernel.org/en/users/drivers/ath5k/devices`, [2018, February 20], 2015.

[75] External products sold which can use ath9k [Online]: Available at: `https://wireless.wiki.kernel.org/en/users/drivers/ath9k/products/external`, [2018, February 20], 2015.

[76] Comparison of open-source wireless drivers [Online]: Available at: `https://en.wikipedia.org/wiki/Comparison_of_open-source_wireless_drivers`, [2018, February 20], 2018.

[77] Llorca, D.F., Sotelo, M.A., Sánchez, S., Ocaña, M., Rodríguez-Ascariz, J.M. and García-Garrido, M.A.: Traffic Data Collection for Floating Car Data Enhancement in V2I Networks. *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, pp. 1–13, 2010.

[78] Khaitiyakun, N. and Sanguankotchakorn, T.: An Analysis of Data Dissemination on VANET by using Content Delivery Network (CDN) technique. In: *Proceedings of the AINTEC 2014 on Asian Internet Engineering Conference - AINTEC '14*, pp. 37–42. ACM Press, New York, New York, USA, 2014. ISBN 9781450332514.

[79] Piñol, P., López, O., Martínez, M., Oliver, J. and Malumbres, M.P.: Modeling video streaming over VANETs. In: *Proceedings of the 7th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks - PM2HW2N '12*, p. 7. ACM Press, New York, New York, USA, 2012. ISBN 9781450316262.

[80] Rezende, C., Pazzi, R.W. and Boukerche, A.: A reactive solution with a redundancy-based error correction mechanism for video dissemination over vehicular ad hoc networks. In: *Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '12*, p. 343. ACM Press, New York, New York, USA, 2012. ISBN 9781450316286.

[81] M. Al-Tahrawi, M.A., Ismail, M., Nordin, R. and Yuwono, T.: Performance of AODV and OLSR routing protocol in a hybrid sensor and vehicular network 802.11p. In: *2016 2nd International Conference on Wireless and Telematics (ICWT)*, pp. 140–145. IEEE, 2016. ISBN 978-1-5090-2649-4.

[82] Noori, H. and Valkama, M.: Impact of VANET-based V2X communication using IEEE 802.11p on reducing vehicles traveling time in realistic large scale urban area. In: *2013 International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 654–661. IEEE, 2013. ISBN 978-1-4799-2491-2.

[83] Saini, M. and Singh, H.: Vanet its characteristics attacks and routing techniques: A survey. *International Journal of Science and Research*, vol. 5, no. 5, pp. 1595–1599, 2016.

[84] Jiang, D., Chen, Q. and Delgrossi, L.: Optimal data rate selection for vehicle safety communications. In: *Proceedings of the Fifth ACM International Workshop on VehiculAr Inter-NETworking*, VANET '08, pp. 30–38. ACM, New York, NY, USA, 2008. ISBN 978-1-60558-191-0.
Available at: `http://doi.acm.org/10.1145/1410043.1410050`

[85] Path Loss Models[Online]: Available at: `https://inet.omnetpp.org/docs/showcases/wireless/pathloss/doc/index.html`, [2018, June 23], 2018.

[86] Angeles, W., Borin, V.P., Munaretto, A. and Fonseca, M.: The Impact of Propagation Models in the Performance of Ad Hoc Routing Protocols for Urban

VANET. In: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5. IEEE, 2016. ISBN 978-1-5090-1701-0.

[87] Perkins, C., Belding-Royer, E. and Das, S.: Ad hoc On-Demand Distance Vector (AODV) Routing. Tech. Rep., jul 2003.
Available at: https://www.rfc-editor.org/info/rfc3561

[88] Lisov, R., Sojka, M. and Hanz, Z.: IEEE 802.11p Linux Kernel Implementation. Tech. Rep., Czech Technical University in Prague, 2014.
Available at: https://rtime.felk.cvut.cz/publications/public/ieee80211p{_}linux{_}2014{_}final{_}report.pdf

[89] CTU-IIG/802.11p-linux [Online]: Available at: https://github.com/CTU-IIG/802.11p-linux, [2018, March 2], 2014.

[90] lisovy/802.11p-on-linux [Online]: Available at: https://gist.github.com/lisovy/80dde5a792e774a706a9, [2018, March 4], 2014.

[91] AODV-UU [Online]: Available at: https://github.com/erimatnor/aodv-uu, [2018, January 13], 2011.

[92] Chakeres, I.D. and Belding-Royer, E.M.: AODV Implementation Design and Performance Evaluation. Tech. Rep..
Available at: http://people.cs.ucsb.edu/ebelding/sites/people/ebelding/files/publications/ijwmc05.pdf

[93] AODV-UIUC[Online]: Available at: https://sourceforge.net/projects/aslib/files/AODV-UIUC/, [2018, January 13], 2002.

[94] Höfner, P., van Glabbeek, R.J., Tan, W.L., Portmann, M., McIver, A. and Fehnker, A.: A rigorous analysis of AODV and its variants. In: *Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '12*, p. 203. ACM Press, 2012. ISBN 9781450316286.
Available at: http://dl.acm.org/citation.cfm?doid=2387238.2387274

[95] Full-size PCIe MINI CARD WLE200NX[Online]: Available at: https://www.pcengines.ch/pdf/wle200nx.pdf, [2018, March 13], 2010.

[96] Wang, Y., Duan, X., Tian, D., Lu, G. and Yu, H.: Throughput and Delay Limits of 802.11p and its Influence on Highway Capacity. *Procedia - Social and Behavioral Sciences*, vol. 96, pp. 2096–2104, nov 2013. ISSN 1877-0428.
Available at: https://www.sciencedirect.com/science/article/pii/S1877042813023628

[97] Song, Y.-S. and Choi, H.-K.: Analysis of V2V Broadcast Performance Limit for WAVE Communication Systems Using Two-Ray Path Loss Model. *ETRI Journal*, vol. 39, no. 2, pp. 213–221, apr 2017. ISSN 1225-6463.
Available at: http://doi.wiley.com/10.4218/etrij.17.2816.0009

[98] Lukic, V., Vujic, S., Makarov, A. and Popovic, Z.: Stereoscopic vehicle speed measurement - system calibration and synchronization errors analysis. In: *2011 International Conference on 3D Imaging (IC3D)*, pp. 1–6. Dec 2011. ISSN 2379-1772.

Appendices

SUMO Maps and Obstacles Additional Figures

SUMO Map Creation and Trip Generation

Converting OSM File to SUMO Road Network File

NETCONVERT is used to convert the OSM file to a SUMO-road network file. The following parameters were passed to the NETCONVERT utility:

```
netconvert --osm-files map.osm -o map.net.xml
--roundabouts.guess --junctions.join --tls.guess-signals
--tls.discard-simple --tls.join --remove-edges.isolated --lefthand
```

Parameters succeeding the *-o* parameter are optional, but were used to enhance the quality of the generated SUMO-road network. An explanation of the rationale behind each parameter is presented in Table 1.

Table 1: NETCONVERT parameter rationale

Parameter	Function
–roundabouts.guess	Applies right-of-way rules to roundabouts since it is not imported from OSM. Roundabouts are a key part of central Stellenbosch
–junctions.join	Merge 'duplicate' junctions to a single intersection that can be represented by a single node
–tls.guess-signals	Interpret 'uncontrolled' intersections which are supposed to be controlled. This is caused by nodes representing traffic lights being placed ahead of intersection nodes. 'Controlled' means to have a traffic light controller present at the intersection.
–tls.join	Create joint traffic light controllers to syncronise traffic lights at intersections
–tls.discard-simple	Recommended parameter, as indicated in NETCONVERT documentation
–remove-edges.isolated	Edges that have no predecessor and successor edges are removed. Not doing so can result in a vehicle being stuck on an isolated road. NETCONVERT documentation describes this option as best to use in vehicle-only scenarios
–lefthand	Applying left-hand driving rules to the road network

Generating Trips/Routes

This is done by using a script obtained in the SUMO source files, which generates trips for a SUMO-road network. Trips are generated at random by uniformly choosing a source and destination edge, and distributed evenly over

a defined interval in time. The result is a XML file containing all the trips.

Generating valid trips with *randomTrips.py* is performed in two steps. First, routes are generated by passing the road network file, desired interval and desired arrival rate to *randomTrips.py* in the following format:

```
randomTrips.py -n network_file.net.xml -o trips_file.trips.xml
-b start_time -e end_time -p arrival_rate_period
```

Interval is the simulation time window (in seconds) in which trips are allowed to start, i.e. vehicles will depart and start their routes between *-b* and *-e* seconds. By default, *-b* is set to 0 seconds. Arrival rate is the constant rate at which vehicles will generated in the network, determined by the period (*-p*) parameter thus $\frac{1}{-p}$ trips will start during each simulation second. In conclusion, for n vehicles to depart between $start\_time$ and end_time, *-p* needs to be set to $\frac{end\_time - start\_time}{n}$. *randomTrips.py* does however not validate weather it is possible for a vehicle to reach its destination from its assigned starting location. Destinations may be unreachable due to parts of the road network that are not connected, this is why a second step is needed. DUAROUTER is a tool which checks for such discontinuities, and validates 'broken' trips. DUAROUTER can be called with *randomTrips.py* in the following format:

```
randomTrips.py -n network_file.net.xml
-r previously_generated_trips_file.trips.xml
-b start_time -e end_time -p arrival_rate_period --validate
```

The *-b*, *-e* and *-p* parameters should be identical to those used in the first step. Since trip generation is random, seeding the algorithms with an optional *-s* parameter is critical when possible adjustments need to be made in the future and the resulting trips need to be identical, i.e. when additional trips need to be appended due to extension of simulation time. In the case of this project, 3454326 was used to seed all trip generations.

Map Feature Extraction

POLYCONVERT was used with the following parameters to create a shape file from the OSM map file:

```
polyconvert --net-file network_file.net.xml --osm-files osm_file.osm
--type-file typemap.xml -o poly_file.poly.xml
```

Additional Map Figures

Figure 1: Area of interest satellite view. Source: `https://www.google.co.za/maps/@-33.9345139,18.864254,5610m/data=!3m1!1e3`

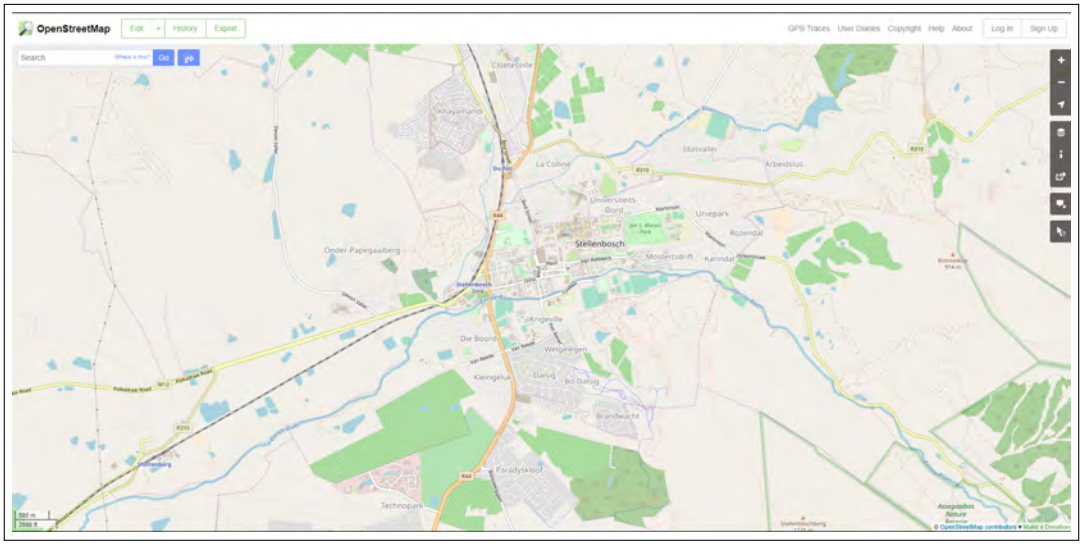


Figure 2: Area of interest in OSM. Source: `https://www.openstreetmap.org/#map=14/-33.9352/18.8646`

Figure 3: Downloaded area of interest in JOSM

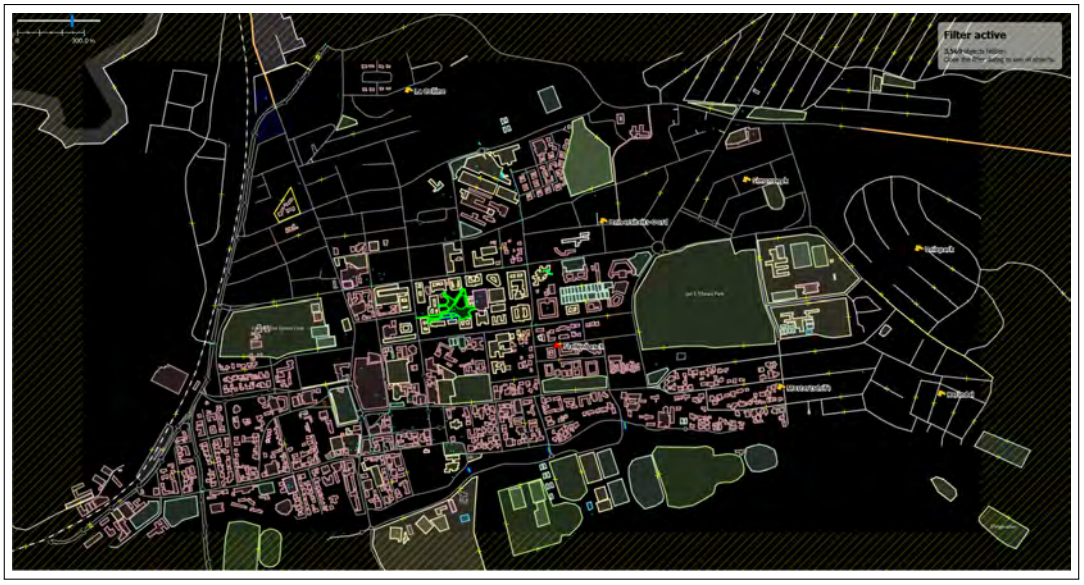


Figure 4: Refined final road network map in JOSM

Figure 5: Refined final road network map in SUMO

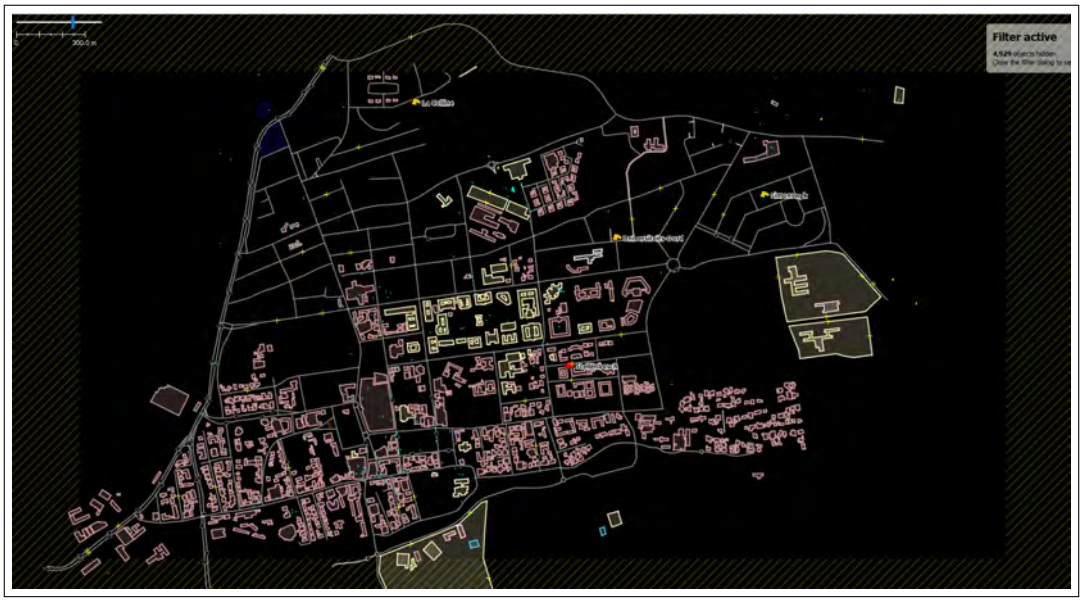


Figure 6: Refined final road network map with buildings in JOSM

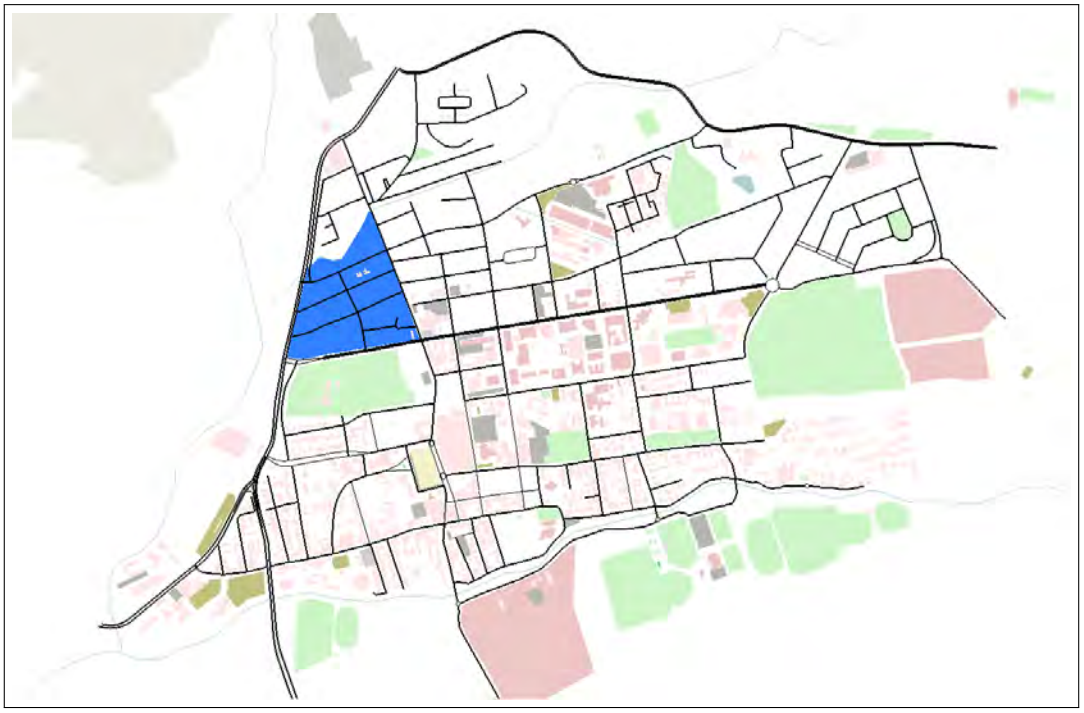


Figure 7: SUMO road network with all visual features

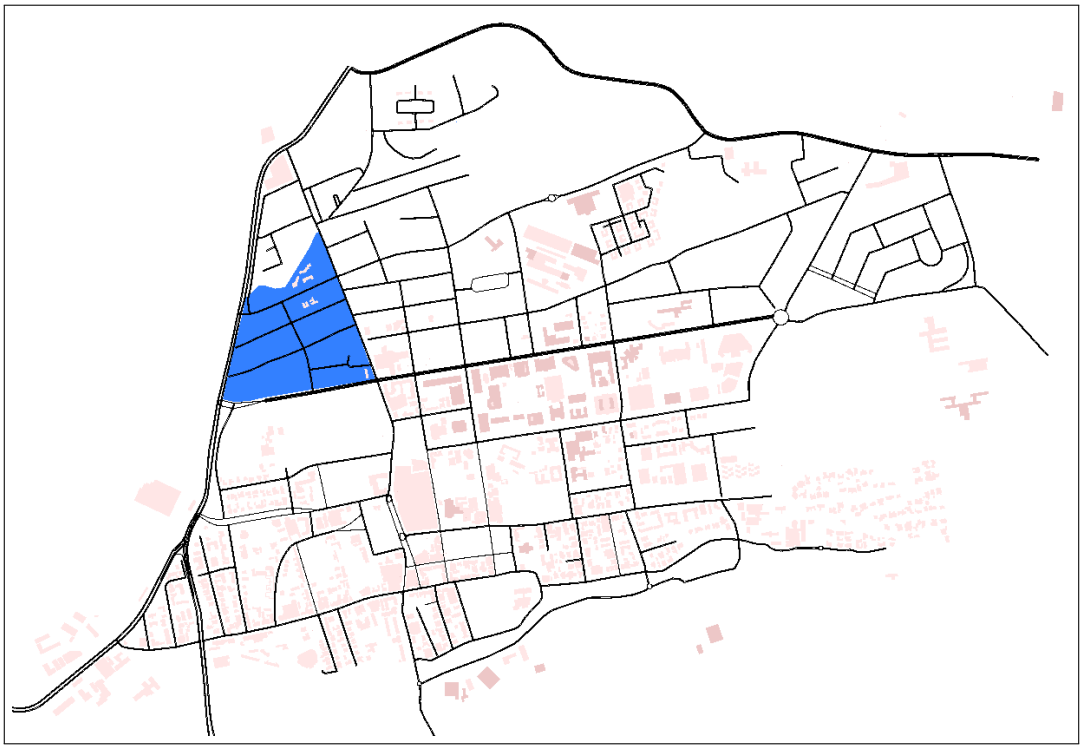


Figure 8: SUMO road network with only buildings and selected amenities

Hardware Implementation Additional Information

Setup of IEEE 802.11p-enabled Linux

The following repositories needs to be added to the /etc/apt/sources.list file due to Ubuntu 15.04 being EOL:

Figure 9: Adding repositories to Ubuntu 15.04

```
1 deb http://old-releases.ubuntu.com/ubuntu/ vivid main restricted
    universe multiverse
2 deb http://old-releases.ubuntu.com/ubuntu/ vivid-updates main
    restricted universe multiverse
3 deb http://old-releases.ubuntu.com/ubuntu/ vivid-security main
    restricted universe multiverse
4
5 deb-src http://old-releases.ubuntu.com/ubuntu/ vivid main restricted
    universe multiverse
6 deb-src http://old-releases.ubuntu.com/ubuntu/ vivid-updates main
    restricted universe multiverse
7 deb-src http://old-releases.ubuntu.com/ubuntu/ vivid-security main
    restricted universe multiverse
```

Setting MAC80211_OCB_DEBUG and CONFIG_MAC80211_STA_DEBUG to TRUE is done by selecting “Verbose OCB debugging" and “Verbose station debugging" respectively, as shown in the figure below.

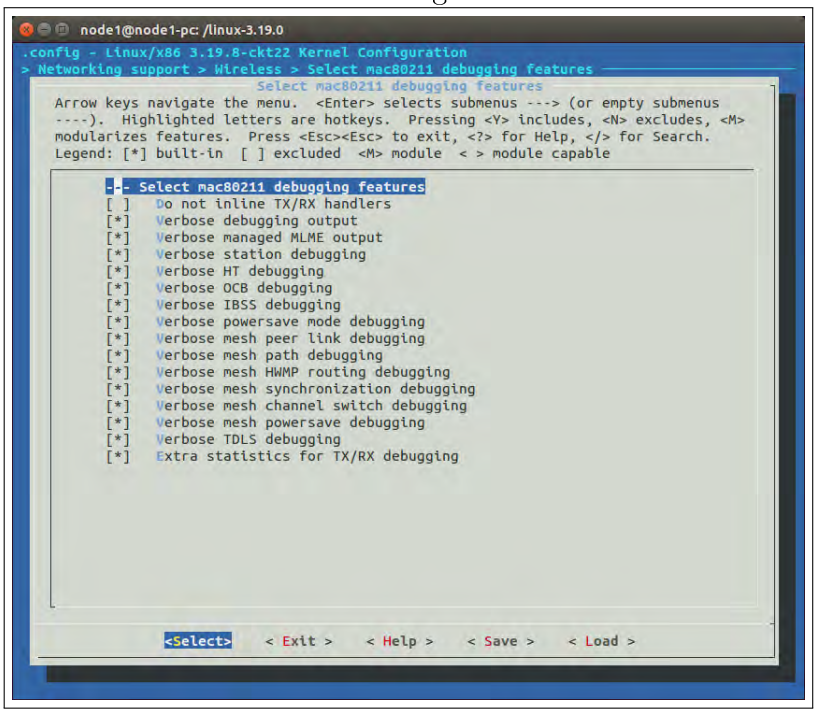


Figure 10: Enabling debugging in kernel configuration

Enabling of the ath9k driver is done as follows:

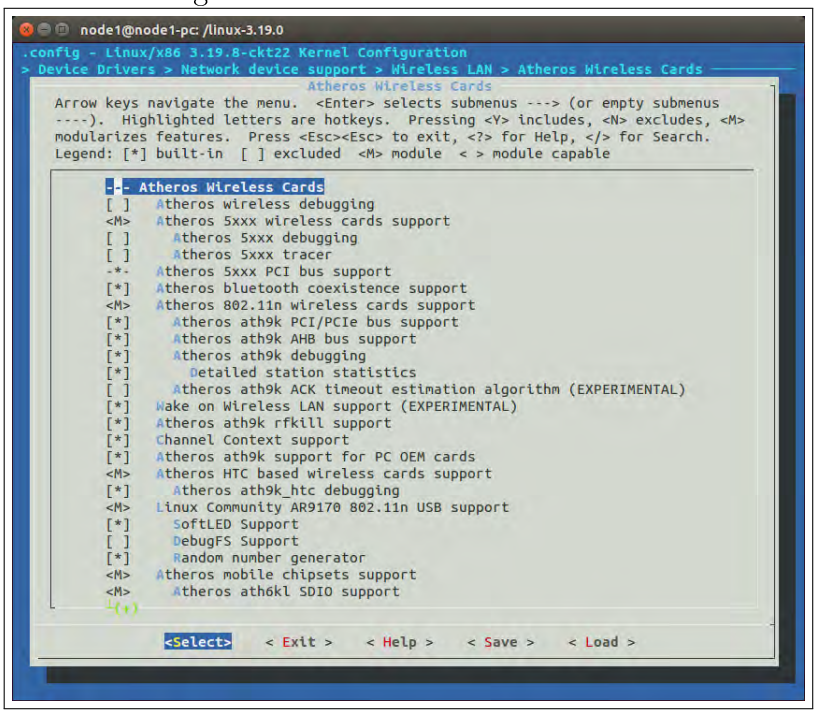


Figure 11: Enabling ATH9K in kernel configuration

The figure below is an extract from `/etc/default/grub`.

```
1 GRUB_TIMEOUT=10
2 GRUB_DISABLE_LINUX_UUID=true
3 GRUB_CMDLINE_LINUX="console=tty1␣console=ttyS0,19200n8␣net.ifnames=0"
4 GRUB_SERIAL_COMMAND="serial␣--speed=19200␣--unit=0␣--word=8␣--parity=
      no␣--stop=1"
5 GRUB_TERMINAL=serial
6 GRUB_GFXPAYLOAD_LINUX=text
7 GRUB_DISABLE_OS_PROBER=true
```

Figure 12: Updating the GRUB

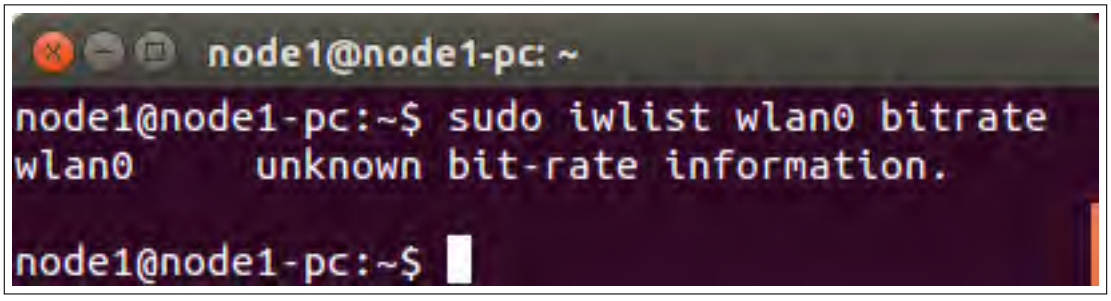


Figure 13: `iwlist` when OCB mode is enabled

Theoretical Throughput Calculation Parameters

Table 2: Theoretical throughput calculation parameter table

Theoretical Throughput Calculation Parameter Values		
Parameter	**Value**	**Description**
Timing Parameters		
T_{pre}	32μs	PLCP preamble duration
T_{ph}	64μs	PHY header transmission duration
T_{sym}	8μs	Symbol time
T_{prop}	1μs	Signal propagation time
T_{DIFS}	58μs	DIFS time
T_{SIFS}	32μs	SIFS time
Data Length Paramters		
L_{data}		Data payload length (Bytes)
L_{dh}	32	Data MAC header (Bytes)
L_{fcs}	4	FCS

Additional Simulation Information and Figures

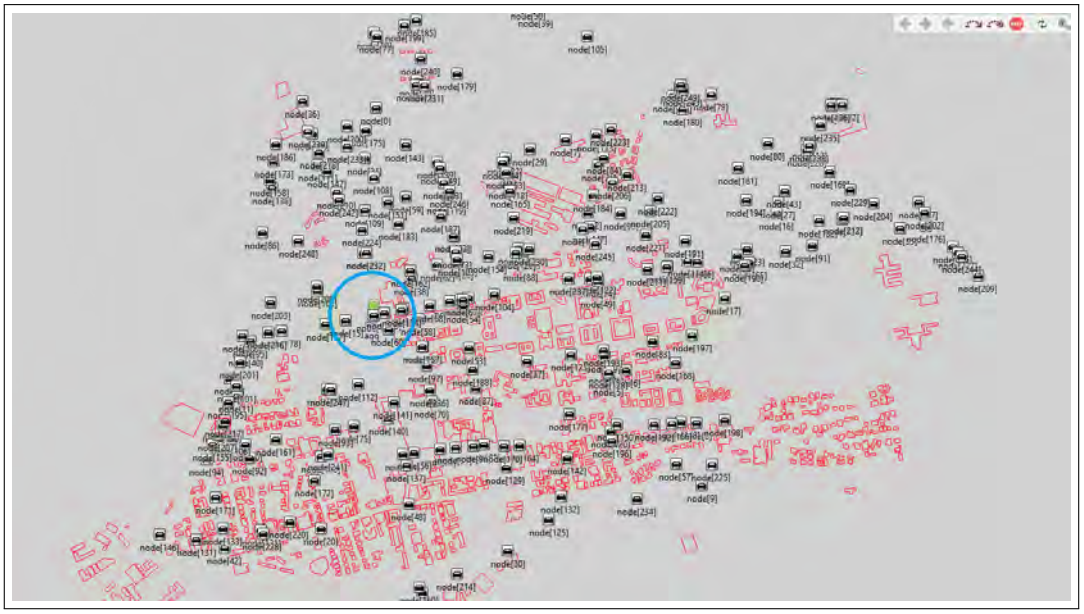


Figure 14: Aggregator placement in actual simulation

Table 3: Final network simulation parameters

Parameter	Value
Environment and Radio Medium Parameters	
*.radioMedium.mediumLimitCache.carrierFrequency	5.86GHz
*.mediumType	Ieee80211ScalarRadioMedium
*.physicalEnvironment.groundType	FlatGround
**.pathLossType	NakagamiFading
*.radioMedium.obstacleLossType	DielectricObstacleLoss
Radio Parameters	
..wlan[0].radioType	Ieee80211ScalarRadio
*. *.wlan[0].radio.transmitter.power	50.12mW (or 79.43mW if unity gain antenna is used)
..wlan[0].radio.receiver.ignoreInterference	true
..wlan[0].radio.receiver.sensitivity	-81dBm
..wlan[0].radio.bandName	"5.9 GHz"
..wlan[0].radio.bandwidth	10MHz
..wlan[0].radio.carrierFrequency	5.86GHz
MAC Parameters	
..wlan[0].macType	Ieee80211Mac
..wlan[0].mgmtType	Ieee80211MgmtAdhoc
..wlan[0].mac.modeSet	"p"
..wlan[0].mac.EDCA	true
..wlan[0].mac.qosStation	true
..wlan[0].mac.bitrate	12Mbps
..wlan[0].mac.basicBitrate	12Mbps
..wlan[0].mac.AIFSN0 to AIFSN3	9,6,3,2
..wlan[0].mac.TXOP0 to TXOP3	0s
Other Parameters	
..wlan[*].opMode	"p"
*.configurator.addStaticRoutes	false
..forwarding	true
..aodv.useHelloMessages	true

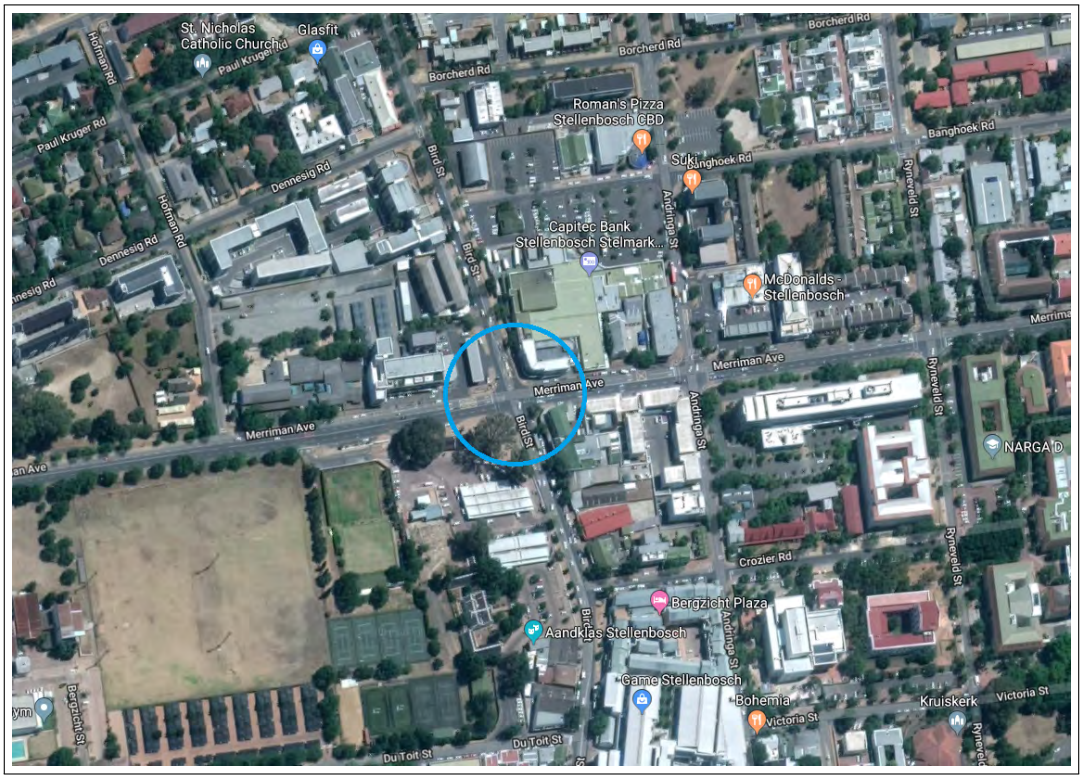


Figure 15: Aggregator placement as per satellite image. Source: `https://www.google.co.za/maps/@-33.9324529,18.8585127,839m/data=!3m1!1e3`